Guia do usuário

AWS Identity and Access Management

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Identity and Access Management: Guia do usuário

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

Table of Contents

O que é o IAM? 1
Por que devo usar o IAM? 3
Acesso compartilhado à sua Conta da AWS 3
Permissões granulares 3
Acesso seguro a recursos da AWS para aplicações executadas no Amazon EC2 4
Autenticação multifator (MFA) 4
Federação de identidades 4
informações de identidade para garantia 4
Compatibilidade com PCI DSS 4
Quando posso usar o IAM? 5
Quando você executa diferentes funções de trabalho 5
Quando você tem autorização para acessar recursos da AWS 6
Quando você fizer login como usuário do IAM 6
Quando você assume um perfil do IAM 6
Quando você cria políticas e permissões 7
Como gerenciar o IAM? 8
Usar a Console de gerenciamento da AWS 8
AWSFerramentas de linha de comando da 10
Usar os SDKs da AWS 12
Usar a API Query do IAM 13
Como o IAM funciona 13
Componentes de uma solicitação 14
Como as entidades principais são autenticados 15
Princípios básicos da política de autorização e permissão 15
..... 16
Comparar identidades e credenciais do IAM 17
Termos 18
Diferenças entre usuários do IAM e usuários do IAM Identity Center 20
Federar usuários a partir de uma fonte de identidades existente 21
Diferentes métodos de fornecer acesso aos usuários 22
Compatibilidade com acesso programático dos usuários 26
Como permissões e políticas fornecem gerenciamento de acesso 27
Políticas e contas 28
Políticas e usuários 28

Políticas e grupos do IAM ... 29
Sessões de usuários federados e perfis ... 29
Políticas baseadas em identidade e em recursos ... 30
Definir permissões com autorização ABAC ... 31
Comparação do ABAC com o modelo RBAC tradicional ... 32
Introdução ... 34
Configurar a Conta da AWS ... 35
Visualizar o ID da Conta da AWS ... 37
Para visualizar sua ID da Conta da AWS ... 37
Usar um alias para seu ID da Conta da AWS ... 39
Criar um alias de conta ... 40
Excluir um alias de conta ... 41
Planeje o acesso à sua conta da AWS ... 42
Fontes de identidade ... 42
Gerenciamento de acesso ... 43
Casos de uso de usuários do IAM ... 43
Usar autenticação multifator com suas identidades ... 57
Preparar para permissões de privilégio mínimo ... 58
Revisar informações acessadas por último da sua conta da AWS ... 59
Gerar uma política com base na atividade de acesso ... 65
Usar pesquisa para encontrar recursos do IAM ... 69
Melhores práticas e casos de uso de segurança ... 73
Práticas recomendadas de segurança ... 73
Exija que os usuários humanos usem a federação com um provedor de identidade para acessar a AWS usando credenciais temporárias ... 74
Exija que as workloads usem credenciais temporárias com perfis do IAM para acessar a AWS ... 75
Exija autenticação multifator (MFA) ... 76
Atualize as chaves de acesso quando necessário para casos de uso que exijam credenciais de longo prazo ... 77
Siga as melhores práticas para proteger as credenciais do usuário-raiz ... 78
Aplique permissões de privilégio mínimo ... 78
Comece com as políticas gerenciadas pela AWS e avance para as permissões de privilégio mínimo. ... 78
Use o IAM Access Analyzer para gerar políticas de privilégios mínimos com base na atividade de acesso ... 79

Revise e remova regularmente usuários, funções, permissões, políticas e credenciais não utilizados 79
Use condições nas políticas do IAM para restringir ainda mais o acesso 79
Verifique o acesso entre contas e público aos recursos com o IAM Access Analyzer 80
Use o analisador de acesso do IAM para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais 80
Estabeleça barreiras de proteção para permissões em várias contas 80
Use limites de permissões para delegar o gerenciamento de permissões em uma conta 81
As práticas recomendadas do usuário-raiz 81
Proteja suas credenciais de usuário-raiz para evitar o uso não autorizado 83
Use uma senha de usuário-raiz forte para ajudar a proteger o acesso 83
Proteja o acesso do seu usuário-raiz com autenticação multifator (MFA) 83
Não crie chaves de acesso para o usuário-raiz 84
Utilize aprovação por múltiplas pessoas para o login do usuário-raiz sempre que possível 84
Use um endereço de e-mail do grupo para as credenciais do usuário-raiz 85
Restrinja o acesso aos mecanismos de recuperação de conta 85
Proteja as credenciais de usuário-raiz da sua conta do AWS Organizations 86
Monitore o acesso e o uso 87
Casos de uso de negócios 89
Configuração inicial da Exemplo Corp 89
Caso de uso do IAM com o Amazon EC2 90
Caso de uso do IAM com o Amazon S3 92
Tutoriais 95
Delegar acesso entre Contas da AWS usando perfis 95
Considerações 97
Pré-requisitos 98
Para criar um perfil na conta Destino 98
Conceder acesso ao perfil 102
Teste o acesso alternando funções 104
Recursos adicionais 110
Resumo 110
Criar uma política gerenciada pelo cliente 110
Pré-requisitos 111
Etapa 1: Criar a política 111
Etapa 2: Anexar a política 112
Etapa 3: Testar o acesso do usuário 113

Recursos relacionados 113
Resumo 113
Usar controle de acesso baseado em atributos (ABAC) 114
Visão geral do tutorial 115
Pré-requisitos 116
Etapa 1: Criar usuários de teste 117
Etapa 2: Criar a política de ABAC 119
Etapa 3: Criar funções 123
Etapa 4: Testar a criação de segredos 125
Etapa 5: Testar a visualização de segredos 129
Etapa 6: Testar a escalabilidade 131
Etapa 7: Testar a atualização e a exclusão de segredos 133
Resumo 134
Recursos relacionados 135
Usar tags de sessão SAML para ABAC 135
Permitir que os usuários gerenciem suas credenciais e configurações de MFA 140
Pré-requisitos 141
Etapa 1: Criar uma política para impor o login com MFA 142
Etapa 2: Anexar políticas ao grupo de usuários de teste 143
Etapa 3: Testar o acesso do usuário 144
Recursos relacionados 146
Criar o IdP do SAML com o CloudFormation 146
Pré-requisitos 146
Criar o IdP 147
Verificar o IdP 150
Excluir recursos. 150
Detalhes do modelo 151
Modelo 151
Crie o perfil federado do SAML com o CloudFormation 155
Pré-requisitos 156
Crie o perfil federado 156
Teste o perfil federado 158
Excluir recursos. 159
Detalhes do modelo 159
Modelo 160
Criar o IdP do SAML e perfil federado com CloudFormation 166

Pré-requisitos 167
Criar os recursos da 167
Teste de federação 171
Excluir recursos. 171
Detalhes do modelo 172
Modelo 172
Identidades 182
Usuário-raiz do IAM 183
Usuários do IAM 183
Grupos de usuários do IAM 183
Perfis do IAM 183
Usuário raiz da conta da AWS 183
Gerencie centralmente o acesso raiz para contas-membro 184
Centralização de acesso raiz 185
Execução de uma tarefa privilegiada 189
MFA para o usuário-raiz 196
Alterar a senha 205
Redefinir uma senha de usuário-raiz perdida ou esquecida 207
Criar chaves de acesso para o usuário-raiz 209
Excluir chaves de acesso para o usuário-raiz 211
Tarefas que exigem credenciais de usuário-raiz 213
Recursos adicionais 215
Usuários 216
Como a AWS identifica um usuário do IAM 216
Usuários do IAM e credenciais 217
Usuários do IAM e permissões 219
Usuários do IAM e contas 219
Usuários do IAM como contas de serviço 219
Como os usuários do IAM fazem login na AWS 220
Criar um usuário 225
Visualizar usuários do IAM 227
Renomear um usuário 230
Remover um usuário 232
Controlar o acesso do usuário ao console 237
Alterar permissões de usuário 238
Senhas de usuários 247

Gerenciar chaves de acesso ... 268
Autenticação multifator ... 299
Credenciais específicas do serviço ... 382
Encontrar credenciais não utilizadas ... 392
Gerar relatórios de credenciais ... 397
Credenciais do IAM para o CodeCommit ... 404
Gerenciar certificados de servidor ... 407
User groups (Grupos de usuários) ... 413
Criar grupos do IAM ... 415
Visualizar grupos do IAM ... 417
Editar usuários em grupos do IAM ... 419
Anexar uma política a um grupo de usuários ... 420
Renomear um grupo de usuários ... 421
Excluir um grupo do IAM ... 423
Perfis ... 425
Quando criar um usuário do IAM (em vez de uma função) ... 426
Termos e conceitos das funções ... 427
Recursos adicionais ... 431
O problema "confused deputy" ... 432
Cenários comuns ... 443
Criação de perfil ... 457
Gerenciamento de perfis ... 526
Métodos para assumir um perfil ... 564
Federação e provedores de identidade na AWS ... 726
Federação com o IAM Identity Center ... 728
Federação com o IAM ... 728
Federação com bancos de identidades do Amazon Cognito ... 729
Recursos adicionais ... 729
Cenários comuns ... 730
Federação OIDC ... 736
Federação SAML 2.0 ... 762
Federação de identidades da AWS para serviços externos ... 806
Casos de uso comuns ... 807
Como funciona ... 808
Conceitos básicos da federação de identidades de saída ... 808
Análise de reivindicações de tokens ... 815

Controle do acesso com políticas do IAM ... 821
Credenciais de segurança temporárias ... 823
AWS STS e regiões da AWS ... 824
Cenários comuns para credenciais temporárias ... 824
Amostra de aplicações que usam credenciais temporárias ... 826
Recursos adicionais para credenciais temporárias ... 827
Compare credenciais do AWS STS ... 828
Tokens de portador de serviço ... 832
Solicitar credenciais de segurança temporárias ... 833
Usar credenciais temporárias com recursos da AWS ... 844
Permissões de credenciais de segurança temporárias ... 849
Gerenciar o AWS STS em uma Região da AWS ... 889
AWS STSRegiões e endpoints do ... 894
Habilitar o acesso do intermediador de identidades personalizado ao console ... 899
Tags para recursos do IAM ... 915
Escolher uma convenção de nomenclatura de tag da AWS ... 916
Regras para etiquetar no IAM e no AWS STS ... 918
Marcar usuários do IAM ... 920
Marcar perfis do IAM ... 924
Marcar políticas gerenciadas pelo cliente ... 927
Marcar provedores de identidades OIDC ... 930
Marcar provedores de identidades SAML do IAM ... 933
Marcar perfis de instância ... 936
Marcar certificados do servidor ... 939
Marcar dispositivos com MFA virtuais ... 941
Marcar instâncias do gerenciador de acesso à conta ... 944
Passar tags de sessão ... 946
Gerenciamento de acesso ... 963
Recursos de gerenciamento de acesso ... 964
Políticas e permissões ... 965
Tipos de políticas ... 965
Políticas e o usuário-raiz ... 973
Visão geral das políticas de JSON ... 973
Conceder privilégio mínimo ... 978
Políticas gerenciadas e em linha ... 980
Perímetro de dados ... 991

Limites de permissões 999
Identidade versus recurso 1013
Controlar acesso usando políticas 1016
Controlar o acesso a usuários e funções do IAM usando etiquetas 1029
Controlar o acesso a recursos da AWS usando tags 1032
Acesso a recursos entre contas 1038
Sessões de acesso direto 1045
Exemplo de políticas 1048
Gerenciar políticas do IAM 1133
Recursos adicionais 1133
Criar políticas do IAM 1134
Validação de políticas 1145
Teste de políticas 1146
Adicionar ou remover permissões de identidade 1160
Versionamento de políticas do IAM 1173
Editar políticas do IAM 1177
Excluir políticas do IAM 1188
Refinar permissões usando informações de acesso 1195
Resumos de políticas 1784
Resumo da política (lista de serviços) 1785
Níveis de acesso em resumos de política 1796
Resumo do serviço (lista de ações) 1799
Resumo da ação (lista de recursos) 1805
Exemplo de resumos de políticas 1809
Permissões necessárias para acessar recursos do IAM 1819
Permissões para administração de identidades do IAM 1819
Permissões para trabalhar no Console de gerenciamento da AWS 1821
Conceder permissões entre contas da AWS 1822
Permissões para um serviço acessar outro 1822
Ações necessárias 1823
Exemplos de políticas do IAM 1824
Delegação temporária 1828
Visão geral 1828
Como funciona a delegação temporária 1829
Iniciar uma solicitação de delegação temporária 1831
Analisar as solicitações de delegação temporária 1834

Revogar o acesso temporário da delegação ... 1836
Gerenciamento de permissões para solicitações de delegação ... 1837
Notificações ... 1838
CloudTrail ... 1839
Delegação temporária para parceiros ... 1841
Visão geral ... 1841
Como funciona ... 1829
Qualificação de parceiros ... 1842
Processo de integração ... 1842
Questionário de parceiro ... 1843
Noções básicas sobre a integração ... 1845
Análise das permissões ... 1849
Diretrizes da avaliação de políticas ... 1851
Modelos de políticas ... 1854
Criação da sua integração ... 1865
Gerenciador de acesso à conta ... 1874
O que é gerenciador de acesso à conta? ... 1874
Por que usar o gerenciador de acesso à conta? ... 1874
Capacidades gerais ... 1876
Introdução ... 1877
Pré-requisitos e considerações ... 1878
Habilitar o gerenciador de acesso à conta ... 1878
Atualizar firewalls e gateways ... 1880
Excluir gerenciador de acesso à conta ... 1881
Gerenciar o acesso ao Contas da AWS ... 1883
Administradores delegados ... 1883
Preparar os perfis do IAM ... 1886
Atribuir e remover acesso ... 1888
Exibir ou pesquisar atribuições ... 1893
Sessões e controle de acesso ... 1895
Acesso da força de trabalho ... 1896
Portais de usuários ... 1896
Conceitos básicos do portal de acesso à conta ... 1897
Acesso dos usuários a Contas da AWS ... 1898
Segurança ... 1903
Gerenciamento de identidade e acesso para o gerenciador de acesso à conta ... 1903

Proteção de dados no gerenciador de acesso à conta ... 1912
Geração de logs e monitoramento no gerenciador de acesso à conta ... 1912
Cotas ... 1912
Cota de atribuição de grupo ... 1912
Limites de controle de utilização ... 1913
Exemplos de código ... 1914
IAM ... 1921
Conceitos básicos ... 1927
Cenários ... 2727
AWS STS ... 3220
Conceitos básicos ... 3222
Cenários ... 3254
Segurança ... 3507
AWSCredenciais de segurança de ... 3508
Considerações sobre segurança ... 3509
Acesso programático ... 3511
AWSDiretrizes de auditoria de segurança da ... 3515
Quando realizar uma auditoria de segurança ... 3516
Diretrizes para a auditoria ... 3516
Analisar as credenciais da sua conta da AWS ... 3516
Revisar seus usuários do IAM ... 3517
Revisar seus grupos do IAM ... 3517
Revisar seus perfis do IAM ... 3518
Revisar seus provedores do IAM para SAML e OpenID Connect (OIDC) ... 3518
Analisar os aplicativos móveis ... 3518
Dicas para revisar políticas do IAM ... 3519
Proteção de dados ... 3521
Criptografia de dados no IAM e no AWS STS ... 3522
Gerenciamento de chaves no IAM e no AWS STS ... 3522
Privacidade do tráfego entre redes no IAM e no AWS STS ... 3522
Registrar em log e monitoramento ... 3523
Registrar eventos em log no CloudTrail ... 3524
Rastreamento de tarefas privilegiadas no CloudTrail ... 3552
Validação de conformidade ... 3556
Resiliência ... 3556
Práticas recomendadas para a resiliência do IAM ... 3559

Segurança da infraestrutura 3559
Análise de configuração e vulnerabilidade 3560
AWSPolíticas gerenciadas pela 3560
IAMReadOnlyAccess 3561
IAMUserChangePassword 3561
IAMAccessAnalyzerFullAccess 3561
IAMAccessAnalyzerReadOnlyAccess 3563
AccessAnalyzerServiceRolePolicy 3564
IAMAuditRootUserCredentials 3565
IAMCreateRootUserPassword 3567
IAMDeleteRootUserCredentials 3568
S3UnlockBucketPolicy 3570
SQSUnlockQueuePolicy 3571
AccountAccessManagerServiceRolePolicy 3572
Atualizações da política 3572
Recursos de segurança fora do IAM 3579
IAM Access Analyzer 3582
Identificar recursos compartilhados com uma entidade externa 3582
Identificar acesso interno a recursos críticos para os negócios 3585
Identificação de acessos não utilizados concedidos a perfis e usuários do IAM 3586
Validação de políticas em relação às práticas recomendadas da AWS 3586
Validar políticas de acordo com seus padrões especificados de segurança 3587
Geração de políticas 3587
Preços do IAM Access Analyzer 3587
Descobertas 3588
Como as descobertas funcionam 3590
Conceitos básicos do IAM Access Analyzer 3595
Painel de descobertas 3613
Revisar descobertas 3618
Filtrar descobertas 3621
Arquivar descobertas 3628
Resolver descobertas 3629
Descobertas de erros 3632
Tipos de recursos compatíveis 3637
Administrador delegado 3646
Regras de arquivamento 3649

Monitoramento com o EventBridge ... 3651
integração do CSPM do Security Hub ... 3663
Registrar em log com o CloudTrail ... 3671
Chaves de filtro ... 3674
Usar perfis vinculados a serviço ... 3691
Pré-visualizar o acesso ... 3694
Pré-visualização de acesso no console do Amazon S3 ... 3694
Pré-visualização de acesso com APIs do IAM Access Analyzer ... 3696
Verificações de validação de políticas ... 3700
Como funcionam as verificações de políticas personalizadas ... 3700
Exemplos de políticas de referência para verificar novos acessos ... 3702
Inspecionar verificações de políticas personalizadas com falha ... 3703
Validar com verificações básicas de políticas ... 3703
Referência de verificação de política ... 3706
Validar com verificações de políticas personalizadas ... 3852
Geração de política do IAM Access Analyzer ... 3854
Como funciona a geração de políticas ... 3855
Informações em nível de serviço e ação ... 3855
Coisas a saber ... 3856
Permissões obrigatórias ... 3857
Gerar uma política com base na atividade do CloudTrail (console) ... 3861
Gerar uma política usando dados do AWS CloudTrail em outra conta ... 3865
Gerar uma política com base na atividade do CloudTrail (AWS CLI) ... 3868
Gerar uma política com base na atividade do CloudTrail (API da AWS) ... 3869
Serviços de geração de política do IAM Access Analyzer ... 3869
Cotas do IAM Access Analyzer ... 4426
Solucionar problemas do IAM ... 4429
Não consigo fazer login na minha conta da AWS ... 4429
Perdi minhas chaves de acesso ... 4429
As variáveis da política não estão funcionando ... 4430
As alterações que eu faço nem sempre ficam imediatamente visíveis ... 4431
Não estou autorizado a executar: iam:DeleteVirtualMFADevice ... 4432
Como faço para criar usuários do IAM com segurança? ... 4432
Recursos adicionais ... 4433
Mensagens de erro de acesso negado ... 4434

Eu recebo a mensagem de “acesso negado” quando faço uma solicitação a um serviço da AWS 4435
Eu recebo uma mensagem de "acesso negado" quando faço uma solicitação com credenciais de segurança temporárias 4436
Exemplos de acesso negado 4438
Mensagens de erro de acesso recusado com o ID de autorização (versão prévia) 4445
Visão geral 4445
Conceitos e definições 4446
Solucionar problemas usando o Console de Gerenciamento da AWS 4447
Solução de problemas usando a API ou a AWS CLI da AWS 4448
Quando você não receber um ID de autorização 4449
Considerações adicionais 4449
Serviços com suporte 4450
Problemas do usuário-raiz 4450
Políticas do IAM 4451
Solução de problemas usando o editor visual 4453
Solucionar problemas com resumos de políticas 4458
Solução de problemas de gerenciamento de políticas 4468
Solução de problemas de documentos de políticas JSON 4469
Chaves de acesso e chaves de segurança FIDO 4476
Não consigo habilitar minha chave de segurança FIDO 4476
Não consigo fazer login usando minha chave de segurança FIDO 4477
Perdi ou quebrei minha chave de segurança FIDO 4477
Outros problemas 4478
Perfis do IAM 4478
Não consigo assumir uma função 4478
Uma nova função apareceu na minha conta da AWS 4481
Não consigo editar ou excluir um perfil na minha Conta da AWS 4481
Não estou autorizado a executar: iam:PassRole 4482
Por que não é possível assumir uma função com uma sessão de 12 horas? (AWS CLI, API da AWS) 4482
Recebo um erro quando tento alternar funções no console do IAM 4483
Minha função tem uma política que permite que eu execute uma ação, mas recebo a mensagem "access denied (acesso negado)" 4483
O serviço não criou a versão da política padrão da função 4484
Não há caso de uso para uma função de serviço no console 4485

IAM e Amazon EC2 4486
Ao tentar iniciar uma instância, não vejo o perfil na lista de perfis do IAM do console do Amazon EC2 4487
As credenciais na minha instância são da função incorreta 4488
Quando tento chamar `AddRoleToInstanceProfile`, recebo um erro `AccessDenied`. 4488
Amazon EC2: quando tento iniciar uma instância com um perfil, recebo um erro `AccessDenied` 4488
Não é possível acessar as credenciais de segurança temporárias em minha instância do EC2. 4489
O que os erros do documento `info` na subárvore do IAM significam? 4490
IAM e Amazon S3 4491
Como faço para conceder acesso anônimo a um bucket do Amazon S3? 4491
Estou conectado como um usuário-raiz da Conta da AWS. Por que não consigo acessar um bucket do Amazon S3 com minha conta? 4491
Federação SAML 2.0 4492
Resposta do SAML inválida 4493
RoleSessionName é obrigatório 4494
Não autorizado para AssumeRoleWithSAML 4494
Caracteres do RoleSessionName inválidos 4495
Caracteres de identidade-fonte inválidos 4495
Resposta da assinatura inválida 4495
Chave privada inválida 4496
Falha ao remover a chave privada 4496
Falha ao remover a chave privada porque a ID da chave não corresponde a uma chave privada 4496
Falha ao assumir função 4497
Não foi possível analisar os metadados 4497
Não é possível atualizar o provedor de identidade 4497
Não é possível definir o modo de criptografia de declaração como Obrigatório porque nenhuma chave privada foi fornecida 4498
Não é possível adicionar e remover chaves privadas na mesma solicitação 4498
O provedor especificado não existe 4498
DurationSeconds excede MaxSessionDuration 4498
Limite máximo de 2 chaves privadas atingido 4499
A resposta não contém o público necessário 4499
Como o IAM funciona com outros serviços da AWS 4500

Criar recursos do IAM com o CloudFormation 4500
IAM e os modelos do CloudFormation 4501
Saiba mais sobre o CloudFormation 4501
Usar o AWS CloudShell com o IAM 4502
Obter permissões do IAM para o AWS CloudShell 4502
Interagir com o IAM 4503
Como trabalhar com AWS SDKs 4505
Referência 4507
Identifique recursos da AWS com nomes do recurso da Amazon (ARNs) 4507
Formato ARN 4507
Encontrar o formato do ARN de um recurso 4509
Caminhos em ARNs 4509
Identificadores do IAM 4511
Nomes amigáveis e caminhos 4511
ARNs do IAM 4512
Identificadores exclusivos 4520
Noções básicas sobre prefixos exclusivos de IDs 4521
Obter o identificador exclusivo 4522
IAM e cotas do AWS STS 4523
Requisitos de nome do IAM 4523
Cotas de objetos do IAM 4524
Cotas do IAM Access Analyzer 4526
Cotas do IAM Roles Anywhere 4526
Cotas de solicitações do STS 4526
Limites de caracteres do IAM e do STS 4527
Suporte a endpoint de pilha dupla 4532
Suporte para endpoint com o modo dual stack no IAM 4533
Suporte ao endpoint de pilha dupla do AWS STS 4533
Endpoints da VPC de interface 4534
Disponibilidade de endpoint da VPC 4535
Criar um VPC endpoint para o IAM 4537
Criar um VPC endpoint para o AWS STS 4538
Criar um endpoint da VPC para descoberta do OIDC do AWS STS 4538
Controle de acesso ao AWS STS com as políticas de endpoint da VPC 4541
Uso de chaves de condição da VPC para controlar o acesso federado 4549
Serviços compatíveis com o IAM 4556

Serviços compatíveis com o IAM ... 4558
Mais informações ... 4599
AWS Signature versão 4 ... 4603
Como o SigV4 da AWS funciona ... 4604
Como a AWS SigV4 funciona ... 4605
Quando assinar solicitações ... 4606
Por que as solicitações são assinadas ... 4606
Recursos adicionais ... 4607
Elementos da solicitação do SigV4 ... 4607
Métodos de autenticação ... 4610
Crie uma solicitação assinada ... 4617
Exemplos de assinatura de solicitação ... 4630
Solucionar problemas do SigV4 ... 4632
Referência de políticas ... 4638
Referência de elemento JSON ... 4639
Lógica da avaliação de política ... 4752
Gramática das políticas ... 4780
AWSPolíticas gerenciadas pela para funções de trabalho ... 4789
Chaves de condição globais ... 4807
Chaves de condição do IAM ... 4894
Ações, recursos e chaves de condição ... 4949
Recursos ... 4950
Identidades ... 4950
Credenciais (senhas, chaves de acesso e dispositivos com MFA) ... 4950
Políticas e permissões ... 4951
Federação e delegação ... 4951
IAM e outros produtos da AWS ... 4952
Uso do IAM com o Amazon EC2 ... 4952
Uso do IAM com o Amazon S3 ... 4952
Uso do IAM com o Amazon RDS ... 4952
Uso do IAM com o Amazon DynamoDB ... 4953
Práticas de segurança gerais ... 4953
Recursos gerais da ... 4953
Fazer solicitações de consulta HTTP ... 4955
Endpoints ... 4956
HTTPS obrigatório ... 4956

Assinar solicitações de API do IAM 4956
Histórico do documento 4958

O que é o IAM?

Follow us on Twitter

O AWS Identity and Access Management (IAM) é um serviço da Web que ajuda você a controlar o acesso aos recursos da AWS de forma segura. Com o IAM, é possível gerenciar permissões que controlam quais recursos da AWS os usuários poderão acessar. Com o IAM, você controla quem pode se autenticar (fazer login) e quem tem autorização (permissões) para acessar os recursos. O IAM fornece a infraestrutura necessária para controlar a autenticação e autorização de sua Contas da AWS.

Para obter instruções sobre como configurar a Conta da AWS e criar identidades, consulte Conceitos básicos do IAM.

Identidades

Ao criar uma Conta da AWS, você começa com uma identidade de login única chamada usuário-raiz da Conta da AWS, que tem acesso total a todos os recursos e Serviços da AWS. É altamente recomendável não usar o usuário-raiz em tarefas diárias. Consulte as tarefas que exigem credenciais de usuário-raiz em Tarefas que exigem credenciais de usuário-raiz no Guia do usuário do IAM.

Use o IAM para configurar outras identidades além do usuário-raiz, como administradores, analistas e desenvolvedores, e conceda a eles acesso aos recursos de que precisarem para ter sucesso em suas tarefas.

Gerenciamento de acesso

Depois que um usuário é configurado no IAM, ele usa suas credenciais de login para se autenticar na AWS. A autenticação é fornecida combinando-se as credenciais de login com uma entidade principal (usuário do IAM, entidade principal federada do AWS STS, perfil do IAM ou aplicação) na qual a Conta da AWS confia. Em seguida, é feita uma solicitação para conceder acesso aos recursos para a entidade principal. O acesso será concedido em resposta a uma solicitação de autorização se o usuário tiver recebido permissão para o recurso. Por exemplo, ao acessar o console pela primeira vez e se deparar com a página inicial, você não está acessando um serviço específico. Quando você seleciona um serviço, a solicitação de autorização é enviada para esse serviço, e ele verifica se a sua identidade está na lista de usuários autorizados, quais políticas estão sendo aplicadas para controlar o nível de acesso concedido e outras políticas que possam estar em vigor. As solicitações

de autorização podem ser feitas por entidades principais de sua Conta da AWS ou de outra Conta da AWS na qual você confia.

Depois de autorizada, a entidade principal pode agir ou realizar operações com recursos em sua Conta da AWS. Por exemplo, a entidade principal pode iniciar uma nova instância do Amazon Elastic Compute Cloud, modificar a associação ao grupo do IAM ou excluir buckets do Amazon Simple Storage Service.

> **Tip**
>
> O AWS Skill Builder fornece uma introdução em vídeo de 10 minutos ao IAM: Introdução ao AWS Identity and Access Management no site do AWS Skill Builder.

Disponibilidade do serviço

O IAM, como muitos outros produtos da AWS, oferece consistência final. O IAM atinge alta disponibilidade ao replicar dados em vários servidores dentro de data centers da Amazon em todo o mundo. Se uma solicitação para alterar alguns dados for bem-sucedida, a alteração estará comprometida e armazenada com segurança. No entanto, a alteração deverá ser replicada em todo o IAM, o que pode levar algum tempo. Essas alterações incluem a criação ou a atualização de usuários, grupos, funções ou políticas. Recomendamos que você não inclua essas alterações do IAM nos caminhos de código crítico de alta disponibilidade do seu aplicativo. Em vez disso, faça alterações do IAM em uma rotina de inicialização ou de configuração separada que você executa com menos frequência. Além disso, certifique-se de verificar se as alterações foram propagadas antes que os fluxos de trabalho de produção dependam delas. Para obter mais informações, consulte As alterações que eu faço nem sempre ficam imediatamente visíveis.

Informações sobre custo do serviço

O AWS Identity and Access Management (IAM), o Centro de Identidade do AWS IAM e o AWS Security Token Service (AWS STS) são atributo da conta da AWS oferecidos sem custo adicional. Você só será cobrado quando acessar outros produtos da AWS usando seus usuários do IAM ou as credenciais de segurança temporárias do AWS STS.

A análise de acesso externo do IAM Access Analyzer é oferecida sem custo adicional. Porém, você incorrerá em custos de análise de acesso e verificações de políticas de clientes não utilizadas. Para obter uma lista completa de cobranças e preços do IAM Access Analyzer, consulte Preços do IAM Access Analyzer.

Para obter informações sobre preços de outros produtos da AWS, consulte a página de preços da Amazon Web Services.

Integração com outros serviços da AWS

O IAM está integrado a muitos serviços da AWS. Para obter uma lista dos serviços da AWS que funcionam com o IAM e dos atributo do IAM compatíveis com os serviços, consulte AWSServiços da que funcionam com o IAM.

Por que devo usar o IAM?

O AWS Identity and Access Management é uma ferramenta poderosa para gerenciar com segurança o acesso aos seus recursos da AWS. Um dos principais benefícios de usar o IAM é a capacidade de conceder acesso compartilhado à sua da AWS. Além disso, o IAM permite que você atribua permissões granulares, permitindo controlar exatamente quais ações diferentes usuários podem realizar em recursos específicos. Esse nível de controle de acesso é crucial para manter a segurança do seu ambiente da AWS. O IAM também oferece vários outros recursos de segurança. Você pode adicionar a autenticação multifator (MFA) para uma camada extra de proteção e aproveitar a federação de identidades para integrar perfeitamente os usuários da sua rede corporativa ou de outros provedores de identidade. O IAM também se integra ao AWS CloudTrail, fornecendo informações detalhadas de registro e identidade para apoiar os requisitos de auditoria e conformidade. Ao aproveitar esses recursos, você pode ajudar a garantir que o acesso aos seus recursos essenciais da AWS seja rigorosamente controlado e seguro.

Acesso compartilhado à sua Conta da AWS

Você pode conceder permissões a outras pessoas para administrar e usar recursos em sua conta da AWS sem a necessidade de compartilhar sua senha ou chave de acesso.

Permissões granulares

Você pode conceder permissões diferentes a pessoas diferentes para diferentes recursos. Por exemplo, você pode permitir que alguns usuários tenham acesso completo ao Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift e outros produtos da AWS. Para outros usuários, você pode permitir acesso somente leitura a apenas alguns buckets do Amazon S3 ou permissão para administrar apenas algumas instâncias do Amazon EC2 ou para acessar suas informações de faturamento, mas nada mais.

Acesso seguro a recursos da AWS para aplicações executadas no Amazon EC2

Você pode usar recursos do IAM para fornecer credenciais de forma segura às aplicações que são executadas em instâncias do EC2. Essas credenciais fornecem permissões ao aplicativo para acessar outros recursos da AWS. Os exemplos incluem buckets do S3 e tabelas do DynamoDB.

Autenticação multifator (MFA)

Você pode adicionar a autenticação de dois fatores à sua conta e a usuários individuais para proporcionar segurança extra. Com a MFA, você ou seus usuários devem fornecer não apenas uma senha ou chave de acesso para trabalhar com a sua conta, mas também um código de um dispositivo especialmente configurado. Se você já usa uma chave de segurança FIDO com outros serviços e ela tem uma configuração com suporte da AWS, você pode usar WebAuthn para segurança de MFA. Para obter mais informações, consulte Configurações compatíveis com o uso de chaves de acesso e chaves de segurança

Federação de identidades

Você pode permitir que os usuários que já têm senhas em outro lugar, por exemplo, na sua rede corporativa ou com um provedor de identidade de Internet, acessem sua Conta da AWS. Esses usuários recebem credenciais temporárias que estão em conformidade com as recomendações de melhores práticas do IAM. Usar a federação de identidades aumenta a segurança da sua conta da AWS.

informações de identidade para garantia

Se você usar o AWS CloudTrail, receberá registros de log com informações sobre quem fez solicitações de recursos na sua conta. Essas informações são baseadas em identidades do IAM.

Compatibilidade com PCI DSS

O IAM é compatível com o processamento, o armazenamento e a transmissão de dados de cartão de crédito por um comerciante ou um provedor de serviços e foi aprovado como estando em conformidade com o Data Security Standard (DSS – Padrão de Segurança de Dados) da Payment Card Industry (PCI – Indústria de Pagamento com Cartão). Para obter mais informações sobre o PCI DSS, incluindo como solicitar uma cópia do pacote de conformidade com o PCI da AWS, consulte Nível 1 do PCI DSS.

Quando posso usar o IAM?

AWS Identity and Access Management O é um serviço de infraestrutura central que fornece a base para o controle de acesso com base nas identidades da AWS. Você usa o IAM toda vez que acessa sua conta da AWS. A forma como você usa o IAM dependerá das responsabilidades específicas e das funções de trabalho em sua organização. Os usuários do serviço AWS usam o IAM para acessar os recursos da AWS necessários para seu trabalho diário, com os administradores concedendo as permissões apropriadas. Os administradores do IAM, por outro lado, são responsáveis por gerenciar as identidades do IAM e criar políticas para controlar o acesso aos recursos. Qualquer que seja seu perfil, você interage com o IAM sempre que autentica e autoriza o acesso aos recursos da AWS. Isso pode envolver fazer login como usuário do IAM, assumir um perfil do IAM ou aproveitar a federação de identidades para um acesso contínuo. Compreender os vários recursos e casos de uso do IAM é crucial para gerenciar com eficácia o acesso seguro ao seu ambiente da AWS. Quando se trata de criar políticas e permissões, o IAM fornece uma abordagem flexível e granular. É possível definir políticas de confiança para controlar quais entidades principais podem assumir um perfil, além das políticas baseadas em identidade que especificam as ações e os recursos que um usuário ou perfil podem acessar. Configurando essas políticas do IAM, você pode garantir que os usuários e as aplicações tenham o nível adequado de permissões para realizar as tarefas necessárias.

Quando você executa diferentes funções de trabalho

AWS Identity and Access Management O é um serviço de infraestrutura central que fornece a base para o controle de acesso com base nas identidades da AWS. Você usa o IAM toda vez que acessa sua conta da AWS.

O uso do IAM varia dependendo do trabalho que for realizado na AWS.

- Usuário do serviço: se você usar o serviço da AWS para fazer o trabalho, o administrador fornecerá as credenciais e as permissões necessárias. À medida que usar recursos mais avançados para fazer seu trabalho, você poderá precisar de permissões adicionais. Entender como o acesso é gerenciado pode ajudar a solicitar as permissões corretas ao administrador.
- Administrador do serviço: se você for o responsável por um recurso da AWS em sua empresa, provavelmente terá acesso total ao IAM. Cabe a você determinar quais funcionalidades e recursos do IAM os usuários do serviço devem acessar. Envie as solicitações ao administrador do IAM para alterar as permissões dos usuários de serviço. Revise as informações nesta página para compreender os conceitos básicos do IAM.

- Administrador do IAM: se for um administrador do IAM, você gerenciará identidades do IAM e escreverá políticas para gerenciar o acesso ao IAM.

Quando você tem autorização para acessar recursos da AWS

A autenticação é a forma como fazer login na AWS usando suas credenciais de identidade. Você precisa se autenticar como o Usuário raiz da conta da AWS, como um usuário do IAM ou assumindo um perfil do IAM.

É possível fazer login como uma identidade federada usando credenciais de uma fonte de identidade, como o Centro de Identidade do AWS IAM (Centro de Identidade do IAM), autenticação única ou credenciais do Google/Facebook. Para ter mais informações sobre como fazer login, consulte Como fazer login em sua Conta da AWS no Guia do usuário do Início de Sessão da AWS.

Para acesso programático, a AWS oferece um SDK e uma CLI para assinar solicitações criptograficamente. Para ter mais informações, consulte AWS Signature Version 4 para solicitações de API no Guia do usuário do IAM.

Quando você fizer login como usuário do IAM

Um usuário do IAM é uma identidade com permissões específicas para uma única pessoa ou aplicação. É recomendável usar credenciais temporárias, em vez de usuários do IAM com credenciais de longo prazo. Para saber mais, consulte Exigir que os usuários humanos usem a federação com um provedor de identidade para acessar a AWS usando credenciais temporárias no Guia do usuário do IAM.

Um grupo do IAM especifica um conjunto de usuários do IAM e facilita o gerenciamento de permissões para grandes conjuntos de usuários. Para ter mais informações, consulte Casos de uso de usuários do IAM no Guia do usuário do IAM.

Quando você assume um perfil do IAM

Uma perfil do IAM é uma identidade com permissões específicas que oferece credenciais temporárias. É possível assumir um perfil alternando de um usuário para um perfil do IAM (console) ou chamando uma operação de API AWS CLI ou AWS. Para saber mais, consulte Métodos para assumir um perfil no Manual do usuário do IAM.

Os perfis do IAM são úteis para acesso de usuário federado, permissões de usuário do IAM temporárias, acesso entre contas, acesso entre serviços e aplicações em execução no Amazon EC2. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Quando você cria políticas e permissões

Conceda permissões para um usuário ao criar uma política, que é um documento que lista as ações que um usuário pode executar e os recursos afetados por estas ações. Todas as ações ou recursos que não são explicitamente permitidos são negados por padrão. As políticas podem ser criadas e anexadas às entidades principais (usuários, grupos de usuários, perfis assumidos por usuários e recursos).

É possível usar essas políticas com um perfil do IAM:

- Política de confiança: define qual entidade principal pode assumir o perfil e em que condições. A política de confiança é um tipo específico de política baseada em recursos para perfis do IAM. O perfil pode ter apenas um conjunto de políticas de confiança.
- Políticas baseadas em identidade (em linha e gerenciadas): essas políticas definem as permissões que o usuário do perfil pode executar (ou não tem permissão para executar) e em quais recursos.

Use Exemplos de políticas baseadas em identidade do IAM para ajudar a definir permissões para suas identidades do IAM. Depois de encontrar a política necessária, escolha visualizar a política para visualizar o JSON da política. É possível usar o documento de política JSON como um modelo para suas próprias políticas.

> **Note**
>
> Se você estiver usando o Centro de Identidade do IAM para gerenciar seus usuários, atribua conjuntos de permissões no Centro de Identidade do IAM em vez de anexar uma política de permissões a uma entidade principal. Quando você atribui um conjunto de permissões a um grupo ou usuário no Centro de Identidade do AWS IAM, o Centro de Identidade do IAM cria perfis do IAM correspondentes em cada conta e anexa as políticas especificadas no conjunto de permissões a esses perfis. O Centro de Identidade do IAM gerencia o perfil e permite que os usuários autorizados que você definiu assumam o perfil. Se você modificar o conjunto de permissões, o Centro de Identidade do IAM garantirá que as políticas e perfis correspondentes do IAM sejam devidamente atualizados.
> Você também pode usar o gerenciador de acesso à conta — um atributo do IAM que permite atribuir perfis do IAM existentes que você cria e gerencia para usuários e grupos do IAM

Identity Center, dando acesso ao conjunto completo de atributos de perfis do IAM. Você pode usá-lo com os conjuntos de permissões ou isoladamente.
Para obter mais informações sobre o IAM Identity Center, consulte What is IAM Identity Center? (O que é o IAM Identity Center?) no Guia do usuário do Centro de Identidade do AWS IAM.

Como gerenciar o IAM?

O gerenciamento do AWS Identity and Access Management em um ambiente da AWS envolve o aproveitamento de uma variedade de ferramentas e interfaces. O método mais comum é por meio do Console de gerenciamento da AWS, uma interface baseada na Web que permite realizar uma ampla variedade de tarefas administrativas do IAM, desde a criação de usuários e funções até a configuração de permissões.

Para usuários mais confortáveis com as interfaces de linha de comando, o AWS fornece dois conjuntos de ferramentas de linha de comando: o AWS Command Line Interface e o AWS Tools for Windows PowerShell. Isso permite que você emita comandos relacionados ao IAM diretamente do terminal, geralmente com mais eficiência do que navegar no console. Além disso, o AWS CloudShell permite que você execute comandos da CLI ou SDK diretamente do seu navegador, usando as permissões associadas ao login no console.

Além do console e da linha de comando, a AWS oferece kits de desenvolvimento de software (SDKs) para várias linguagens de programação, permitindo que você integre a funcionalidade de gerenciamento do IAM diretamente em suas aplicações. Como alternativa, você pode acessar o IAM programaticamente usando a API de consulta do IAM, que permite emitir solicitações HTTPS diretamente para o serviço. O aproveitamento dessas diferentes abordagens de gerenciamento oferece a flexibilidade de incorporar o IAM em seus fluxos de trabalho e processos existentes.

Usar a Console de gerenciamento da AWS

O Console de gerenciamento da AWS é uma aplicação Web que compreende e se refere a um amplo acervo de consoles de serviço para gerenciar recursos da AWS. Quando você faz login pela primeira vez, vê a página inicial do console. A página inicial fornece acesso a todos os consoles de serviço e oferece um único local para acessar as informações necessárias para executar suas tarefas relacionadas à AWS. Os serviços e aplicações disponíveis para você depois de entrar no console dependem dos recursos da AWS que você tem permissão para acessar. É possível receber permissões para recursos assumindo um perfil, sendo membro de um grupo ao qual permissões

foram concedidas ou recebendo uma permissão explícita. Para uma conta da AWS independente, o usuário raiz ou o administrador do IAM configura o acesso aos recursos. Nas AWS Organizations, a conta de gerenciamento ou o administrador delegado configura o acesso aos recursos.

Se você planeja que as pessoas usem o Console de gerenciamento da AWS para gerenciar recursos da AWS, recomendamos configurar usuários com credenciais temporárias como uma prática recomendada de segurança. Os usuários do IAM que assumiram um perfil, as entidades principais federadas e os usuários no Centro de Identidade do IAM têm credenciais temporárias, enquanto o usuário do IAM e o usuário raiz têm credenciais de longo prazo. As credenciais do usuário raiz fornecem acesso total à Conta da AWS, enquanto outros usuários têm credenciais que fornecem acesso aos recursos concedidos pelas políticas do IAM.

A experiência de login é diferente para os diferentes tipos de usuários do Console de gerenciamento da AWS.

- Os usuários do IAM e o usuário raiz fazem login via URL de login principal da AWS (https://signin.aws.amazon.com). Depois de fazer login, eles têm acesso aos recursos da conta para a qual receberam permissão.

  Para fazer login como usuário raiz, é necessário ter o endereço de e-mail e a senha do usuário raiz.

  Para entrar como usuário do IAM, é necessário ter o número ou o alias da Conta da AWS, o nome de usuário do IAM e a senha do usuário do IAM.

  Recomendamos restringir os usuários do IAM em sua conta a situações específicas que exijam credenciais de longo prazo, como acesso de emergência, e usar o usuário raiz somente para tarefas que exijam credenciais de usuário raiz.

  Para maior conveniência, a página de login da AWS usa um cookie de navegador para lembrar o nome de usuário e as informações da conta do IAM. Na próxima vez que o usuário acessar qualquer página no Console de gerenciamento da AWS, o console usará o cookie para redirecionar o usuário para a página de login da conta.

  Saia do console ao terminar sua sessão para evitar a reutilização do seu login anterior.
- Os usuários do Centro de Identidade do IAM fazem login usando um portal de acesso específico da AWS que é exclusivo para sua organização. Depois de fazer login, eles podem escolher qual conta ou aplicação acessar. Se optarem por acessar uma conta, eles escolherão qual conjunto de permissões desejam usar para a sessão de gerenciamento.

- As entidades principais federadas OIDC e SAML gerenciadas em um provedor de identidade externo vinculado a um login de Conta da AWS usando um portal de acesso corporativo personalizado. Os recursos da AWS disponíveis para os usuários dependem das políticas selecionadas por sua organização.

> **Note**
>
> Para fornecer um nível adicional de segurança, o usuário raiz, os usuários do IAM e os usuários do Centro de Identidade do IAM podem ter a autenticação multifator (MFA) verificada pela AWS antes de conceder acesso aos recursos da AWS. Quando a MFA está habilitada, você também deve ter acesso ao dispositivo de MFA para fazer login.

Para saber mais sobre como diferentes usuários fazem login no console de gerenciamento, consulte Login no Console de gerenciamento da AWS no Guia do usuário de login da AWS.

AWSFerramentas de linha de comando da

Você pode usar as ferramentas da linha de comando da AWS para emitir comandos na linha de comando do seu sistema e realizar tarefas do IAM e da AWS. Usar a linha de comando pode ser mais rápido e mais conveniente do que o console. As ferramentas da linha de comando também são úteis se você quiser criar scripts que realizem tarefas da AWS.

A AWS fornece dois conjuntos de ferramentas de linha de comando: a AWS Command Line Interface (AWS CLI) e o AWS Tools for Windows PowerShell. Para obter informações sobre a instalação e o uso da AWS CLI, consulte o Guia do usuário da AWS Command Line Interface. Para obter informações sobre a instalação e o uso do Tools for Windows PowerShell, consulte o Guia do usuário do Ferramentas da AWS para PowerShell.

Depois de entrar no console, será possível usar a AWS CloudShell partir do seu navegador para executar comandos da CLI ou do SDK. As permissões para acessar recursos da AWS são baseadas nas credenciais que você usou para entrar no console. Dependendo da sua experiência, talvez você considere a CLI um método mais eficiente de gerenciar a Conta da AWS. Para ter mais informações, consulte Usar o AWS CloudShell para operação com o AWS Identity and Access Management.

Interface de linha de comando (CLI) da AWS e kits de desenvolvimento de software (SDKs)

Os usuários do Centro de Identidade do IAM e os usuários do IAM usam métodos diferentes para autenticar suas credenciais quando se autenticam por meio da CLI ou das interfaces de aplicações (APIs) nos SDKs associados.

As credenciais e as configurações estão localizadas em vários locais, como variáveis de ambiente do sistema ou do usuário, arquivos de configuração local da AWS, ou explicitamente declaradas na linha de comando como um parâmetro. Certos locais têm precedência sobre outros.

Tanto o Centro de Identidade do IAM quanto o IAM fornecem chaves de acesso que podem ser usadas com a CLI ou o SDK. As chaves de acesso do Centro de Identidade do IAM são credenciais temporárias que podem ser atualizadas automaticamente e são recomendadas no lugar das chaves de acesso de longo prazo associadas aos usuários do IAM.

Para gerenciar sua Conta da AWS usando a CLI ou o SDK, é possível usar a AWS CloudShell a partir do seu navegador. Se você usa o CloudShell para executar comandos da CLI ou do SDK, é necessário entrar no console primeiro. As permissões para acessar recursos da AWS são baseadas nas credenciais que você usou para entrar no console. Dependendo da sua experiência, talvez você considere a CLI um método mais eficiente de gerenciar a Conta da AWS.

Para o desenvolvimento de aplicações, é possível baixar a CLI ou o SDK para o seu computador e fazer login no prompt de comando ou em uma janela do Docker. Nesse cenário, você configura as credenciais de autenticação e acesso como parte do script da CLI ou da aplicação do SDK. É possível configurar o acesso programático aos recursos de maneiras diferentes, dependendo do ambiente e do acesso disponível para você.

- As opções recomendadas para autenticar o código local com serviço da AWS são o Centro de Identidade do IAM e o IAM Roles Anywhere
- As opções recomendadas para autenticar o código executado em um ambiente da AWS são usar perfis do IAM ou usar as credenciais do Centro de Identidade do IAM.

Quando você inicia sessão pelo portal de acesso do AWS, pode obter credenciais de curto prazo na página inicial em que escolhe seu conjunto de permissões. Essas credenciais têm duração definida e não são atualizadas automaticamente. Se desejar utilizar essas credenciais, após entrar no portal da AWS, escolha a Conta da AWS e, em seguida, escolha o conjunto de permissões. Selecione Linha de comando ou acesso programático para ver as opções que podem ser usadas para acessar

os recursos da AWS de forma programática ou via CLI. Para obter mais informações sobre esses métodos, consulte Obtenção e atualização de credenciais temporárias no Guia do usuário do Centro de Identidade do IAM. Essas credenciais são frequentemente usadas durante o desenvolvimento da aplicação para testar rapidamente o código.

Recomendamos usar as credenciais do Centro de Identidade do IAM, as quais são atualizadas automaticamente ao automatizar o acesso aos seus recursos da AWS. Se você configurou usuários e conjuntos de permissões no Centro de Identidade do IAM, use o comando `aws configure sso` para usar um assistente de linha de comando que ajudará a identificar as credenciais disponíveis para você e armazená-las em um perfil. Para obter mais informações sobre como configurar seu perfil, consulte Configuração do seu perfil com o assistente `aws configure sso` no Guia do usuário da Interface de linha de comando da AWS para a versão 2.

> **Note**
>
> Muitas aplicações de exemplo usam chaves de acesso de longo prazo associadas aos usuários do IAM ou ao usuário raiz. Use as credenciais de longo prazo somente em um ambiente sandbox como parte de um exercício de aprendizado. Analise as alternativas às chaves de acesso de longo prazo e planeje fazer a transição do seu código para usar credenciais alternativas, como credenciais ou perfis do IAM do Centro de Identidade do IAM, o mais rápido possível. Depois de fazer a transição do código, exclua as chaves de acesso.

Para saber mais sobre como configurar a CLI, consulte Instalação ou atualização da versão mais recente da CLI da AWS no Guia do usuário da Interface de linha de comando da AWS para a versão 2 e Credenciais de autenticação e acesso no Guia do usuário da Interface de linha de comando da AWS

Para saber mais sobre como configurar o SDK, consulte Autenticação do Centro de Identidade do IAM no Guia de referência de SDKs e ferramentas da AWS e IAM Roles Anywhere no Guia de referência de SDKs e ferramentas da AWS.

Usar os SDKs da AWS

A AWS fornece SDKs (kits de desenvolvimento de software) que consistem em bibliotecas e códigos de exemplo para várias linguagens de programação e plataformas (Java, Python, Ruby, .NET, iOS, Android, etc.). Os SDKs constituem uma forma conveniente de criar acesso programático para o IAM e a AWS. Por exemplo, os SDKs processam tarefas como assinatura criptográfica de

solicitações, gerenciamento de erros e novas tentativas automáticas de solicitações. Para obter informações sobre os AWS SDKs, incluindo como fazer download deles e instalá-los, consulte a página Ferramentas para a Amazon Web Services.

Usar a API Query do IAM

Você pode acessar o IAM e a AWS de forma programática usando a API Query do IAM, que permite emitir solicitações HTTPS diretamente ao serviço. Ao usar a API Query, é necessário incluir código para assinar digitalmente solicitações usando suas credenciais. Para obter mais informações, consulte Chamar a API do IAM usando solicitações de consulta HTTP e a Referência da API do IAM.

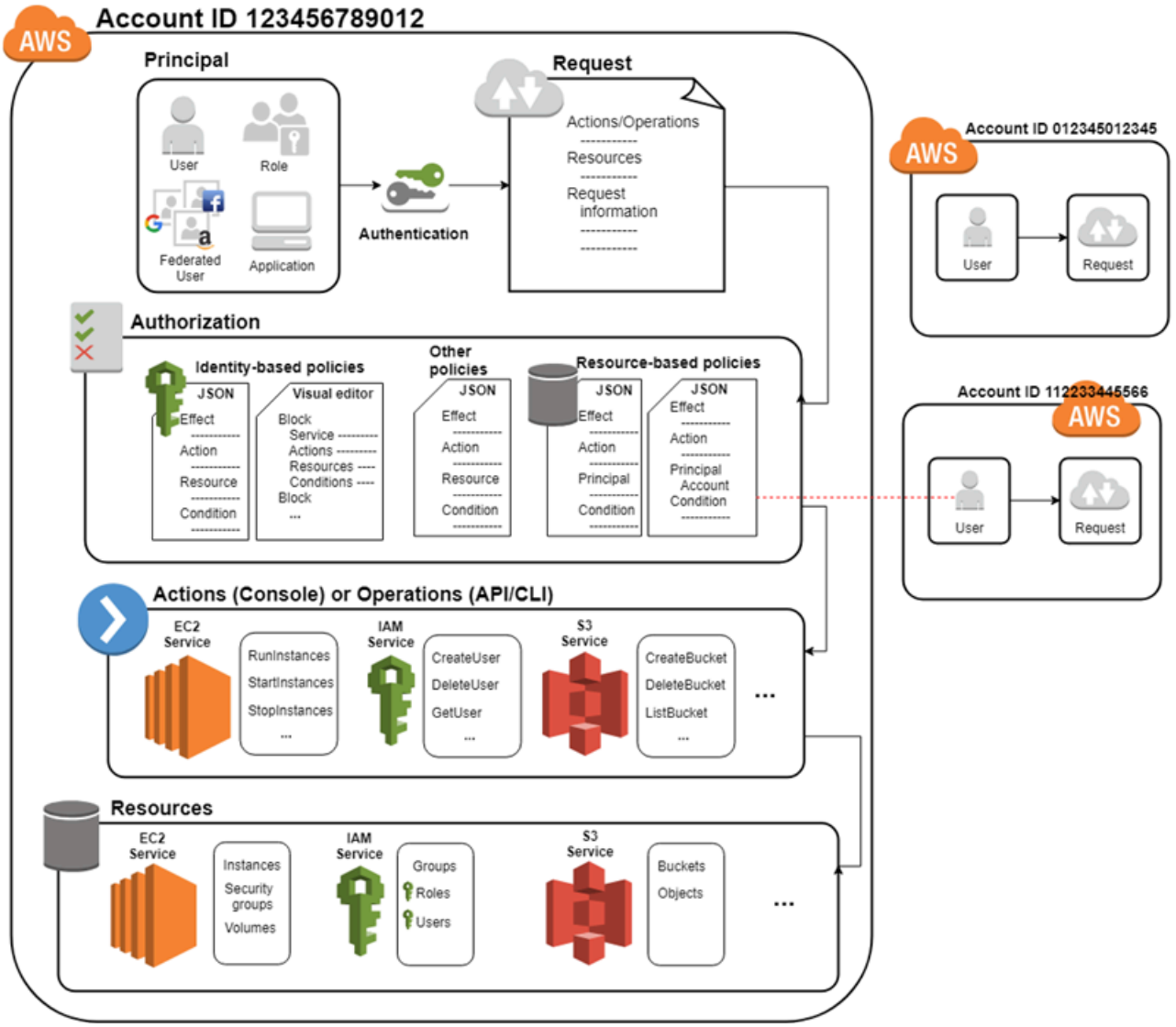
Como o IAM funciona

AWS Identity and Access Management O fornece a infraestrutura necessária para controlar a autenticação e autorização da sua Conta da AWS.

Primeiro, um usuário humano ou uma aplicação usa suas credenciais de login para se autenticar na AWS. O IAM compara as credenciais de entrada com uma entidade principal (um usuário do IAM, uma entidade principal de usuário federado do AWS STS, um perfil do IAM ou uma aplicação) de confiança da Conta da AWS e autentica a permissão para acessar a AWS.

Em seguida, o IAM faz uma solicitação para conceder à entidade principal acesso aos recursos. O IAM concede ou nega o acesso em resposta a uma solicitação de autorização. Por exemplo, ao acessar o console pela primeira vez e ir para a página inicial do console, você não está acessando um serviço específico. Ao selecionar um serviço, você envia ao IAM uma solicitação de autorização para esse serviço. O IAM confirma que a sua identidade está na lista de usuários autorizados, determina quais políticas controlam o nível de acesso concedido e avalia outras políticas que possam estar em vigor. As entidades principais da sua Conta da AWS ou de outra Conta da AWS na qual você confia podem fazer solicitações de autorização..

Uma vez autorizada, a entidade principal pode realizar ações ou operações nos recursos da sua Conta da AWS. Por exemplo, a entidade principal pode iniciar uma nova instância do Amazon Elastic Compute Cloud, modificar a associação ao grupo do IAM ou excluir buckets do Amazon Simple Storage Service. O seguinte diagrama ilustra esse processo ao longo da infraestrutura do IAM:



Componentes de uma solicitação

Quando uma entidade principal tenta usar o Console de gerenciamento da AWS, a API da AWS ou a AWS CLI, ela envia uma solicitação para a AWS. A solicitação inclui as seguintes informações:

- Ações ou operações: as ações ou operações que a entidade principal deseja realizar, como uma ação no Console de gerenciamento da AWS ou uma operação na AWS CLI ou na API da AWS.
- Recursos: o objeto de recurso da AWS no qual a entidade principal deseja realizar ações ou operações.
- Principal – A pessoa ou o aplicativo que usou uma entidade (usuário ou função) para enviar a solicitação. As informações sobre a entidade principal incluem as políticas de permissão.

- Dados do ambiente: informações sobre o endereço IP, o agente do usuário, o status do SSL habilitado e o timestamp.
- Dados do recurso: dados relacionados ao recurso solicitado, como um nome de tabela do DynamoDB ou uma tag em uma instância do Amazon EC2.

O AWS reúne as informações da solicitação em um contexto da solicitação, que o IAM avalia para e autorizá-la.

Como as entidades principais são autenticados

Uma entidade principal entra na AWS usando suas credenciais, que são autenticadas pelo IAM autentica para permitir que a entidade principal envie uma solicitação à AWS. Alguns serviços, como o Amazon S3 e o AWS STS, permitem determinadas solicitações de usuários anônimos. Porém, essas são exceções à regra. Todo tipo de usuário passa pela autenticação.

- Usuário-raiz: as credenciais de entrada usadas para autenticação são o endereço de e-mail que você usou para criar a Conta da AWS e a senha que especificou naquele momento.
- Entidade principal federada: seu provedor de identidades autentica e passa suas credenciais para a AWS; não é necessário entrar diretamente na AWS. Tanto o Centro de Identidade do IAM quanto o IAM são compatíveis com a federação de identidades.
- Usuários no Diretório do Centro de Identidade do AWS IAM (não federados): os usuários criados diretamente no diretório padrão do IAM Identity Center entram usando o portal de acesso do AWS e fornecem o seu nome de usuário e senha.
- Usuário do IAM: você entra fornecendo o ID ou o alias da sua conta, seu nome de usuário e sua senha. Para autenticar workloads usando a API ou a AWS CLI, você pode usar credenciais temporárias assumindo um perfil ou usar credenciais de longo prazo fornecendo sua chave de acesso e sua chave secreta.

  Para saber mais sobre as entidades do IAM, consulte Usuários do IAM e Perfis do IAM.

AWSA recomenda o uso da autenticação multifator (MFA) para aumentar a segurança de sua conta. Para saber mais sobre a MFA, consulte AWSCódigo da autenticação multifator no IAM da.

Princípios básicos da política de autorização e permissão

Autorização se refere à entidade principal ter as permissões necessárias para concluir a solicitação. Durante a autorização, o IAM identifica as políticas que se aplicam à solicitação usando valores do

contexto da solicitação. Em seguida, ela usa as políticas para determinar se deve permitir ou negar uma solicitação. O IAM armazena a maioria das políticas de permissão como documentos JSON que especificam as permissões das entidades principais.

Há vários tipos de políticas que podem afetar uma solicitação de autorização. Para fornecer aos usuários permissões para acessar os recursos da AWS em sua conta, você pode usar políticas baseadas em identidade. As políticas baseadas em recurso podem conceder acesso entre contas. Para fazer uma solicitação em outra conta, uma política nessa outra conta deve permitir que você acesse o recurso e a entidade do IAM que você usa para fazer a solicitação deve ter uma política baseada em identidade que permita a solicitação.

O IAM verifica todas as políticas que se aplicam ao contexto da sua solicitação. A avaliação da política do IAM usa uma negação explícita, o que significa que, se uma única política de permissão incluir uma ação negada, o IAM negará toda a solicitação e interromperá a avaliação. Como as solicitações são negadas por padrão, as políticas de permissões aplicáveis devem permitir que todas as partes da solicitação do IAM autorizem sua solicitação. A lógica de avaliação de uma solicitação em uma única conta segue estas regras básicas:

- Por padrão, todas as solicitações são negadas. (Em geral, as solicitações feitas usando as credenciais Usuário raiz da conta da AWS para recursos na conta são sempre permitidas.)
- Uma permissão explícita em uma política de permissões (baseada em recurso ou identidades) substitui esse padrão.
- A existência de uma política de controle de serviços (SCP) ou de uma política de controle de recursos (RCP) do AWS Organizations, um limite de permissões do IAM ou uma política de sessão se sobrepõe à permissão. Se um ou mais desses tipos de política existir, todos eles devem permitir a solicitação. Se não, ela será implicitamente negada. Para obter mais informações sobre SCPs e RCPs, consulte Políticas de autorização no AWS Organizations no Guia do usuário do AWS Organizations.
- Uma negação explícita em qualquer política anula todas as permissões.

Para saber mais, consulte Lógica da avaliação de política.

Depois que o IAM autentica e autoriza a entidade principal, o IAM aprova as ações ou operações solicitadas avaliando a política de permissão que se aplica à entidade principal. Cada serviço da AWS define com quais ações (operações) ele é compatível e inclui o que você pode fazer com um recurso, como visualizar, criar, editar e excluir esse recurso. A política de permissão aplicável à entidade principal deve incluir as ações necessárias para realizar uma operação. Para saber mais

sobre como o IAM avalia as políticas de permissão, consulte the section called “Lógica da avaliação de política”.

O serviço define um conjunto de ações que uma entidade principal poder realizar em cada recurso. Ao criar políticas de permissão, certifique-se de incluir as ações que você deseja que o usuário possa realizar. Por exemplo, o IAM é compatível com mais de 40 ações para um recurso de usuário incluindo as seguintes ações básicas:

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

Além disso, você pode especificar condições na política de permissão que forneçam acesso aos recursos quando a solicitação atender às condições especificadas. Por exemplo, talvez você queira que uma instrução da política entre em vigor após uma data específica ou controle o acesso quando um valor específico aparecer em uma API. Para especificar condições, use o elemento `Condition` de uma instrução de política.

Depois que o IAM aprova as operações em uma solicitação, a entidade principal pode trabalhar com os recursos relacionados em sua conta. Um recurso é uma objeto que existe dentro de um serviço. Os exemplos incluem uma instância do Amazon EC2, um usuário do IAM e um bucket do Amazon S3. Se a entidade principal criar uma solicitação para realizar uma ação em um recurso que não está incluída na política de permissão, o serviço negará a solicitação. Por exemplo, se tiver permissão para excluir um perfil do IAM, mas solicitar permissão para excluir um grupo do IAM, a solicitação falhará se você não tiver permissão para excluir grupos do IAM. Para saber mais sobre quais ações, recursos e chaves de condição são compatíveis com os diferentes serviços da AWS, consulte Actions, Resources, and Condition Keys for AWS Services.

Comparar identidades e credenciais do IAM

As identidades gerenciadas no AWS Identity and Access Management são usuários, perfis e grupos do IAM. Essas identidades são além do seu usuário-raiz que a AWS criou junto com a sua Conta da AWS.

É recomendável não usar o usuário-raiz para suas tarefas diárias, nem mesmo para as administrativas. Em vez disso, provisione usuários adicionais e conceda-lhes as permissões

necessárias para realizar as tarefas necessárias. É possível adicionar usuários incluindo pessoas no seu diretório do IAM Identity Center, federando um provedor de identidades externo ao IAM Identity Center ou ao IAM ou criando usuários do IAM com privilégio mínimo.

Para ter segurança adicional, recomendamos a centralização do acesso raiz para ajudá-lo a proteger centralmente as credenciais do usuário-raiz das suas Contas da AWS gerenciadas usando o AWS Organizations. Gerencie centralmente o acesso raiz para contas-membro permite que você remova e evite centralmente a recuperação de credenciais de usuário-raiz em longo prazo, evitando o acesso raiz não intencional em grande escala. Depois de habilitar o acesso raiz centralizado, será possível assumir uma sessão privilegiada para realizar ações nas contas-membro.

Depois de configurar seus usuários, você pode dar acesso à sua Conta da AWS a pessoas específicas e lhes fornecer permissões para acessar recursos.

Como prática recomendada, a AWS recomenda o requisito de que usuários humanos assumam um perfil do IAM para acessarem o AWS, de forma que eles usem credenciais temporárias. Se estiver gerenciando identidades no diretório do Centro de Identidade do IAM ou usando federação com um provedor de identidade, você está seguindo as práticas recomendadas.

Termos

Esses termos são usados com frequência quando se trabalha com identidades do IAM:

Recurso do IAM

O serviço IAM armazena esses recursos. Você pode adicioná-los, editá-los e removê-los do Console.

- IAM user (Usuário do IAM)
- IAM group (Grupo do IAM)
- Perfil do IAM
- Política de permissão
- Objeto de provedor de identidades

Entidade do IAM

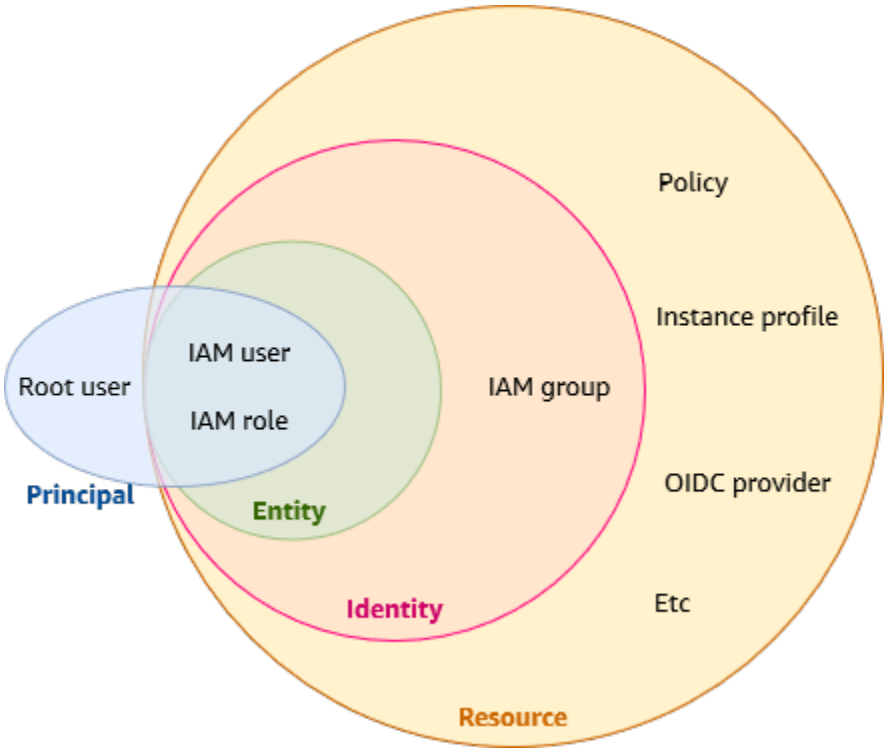
Recursos do IAM que AWS usa para autenticação. Especifique uma entidade principal em uma política baseada em recurso.

- IAM user (Usuário do IAM)

- Perfil do IAM

Identidade do IAM

O recurso do IAM que é autorizado em políticas a realizar ações e acessar recursos. As identidades incluem usuários, grupos e perfis do IAM.



Entidades principais

Um Usuário raiz da conta da AWS, um usuário do IAM ou um perfil do IAM que pode fazer uma solicitação para realizar uma ação ou operação em um recurso da AWS. As entidades principais incluem usuários humanos, workloads, entidades principais federadas e perfis assumidos. Após a autenticação, o IAM concede à entidade principal credenciais permanentes ou temporárias para fazer solicitações à AWS, dependendo do tipo da entidade principal.

Usuários humanos, também conhecidos como identidades humanas, como pessoas, administradores, desenvolvedores, operadores e consumidores de aplicações.

Workloads são um conjunto de códigos e recursos que geram valor comercial, como uma aplicação, um processo, ferramentas operacionais e outros componentes.

Entidades principais federadas são usuários cuja identidade e credenciais são gerenciadas por outro provedor de identidade, como o Active Directory, Okta ou Microsoft Entra.

Perfis do IAM são uma identidade do IAM que você pode criar em sua conta com permissões específicas que determinam o que a identidade pode ou não fazer. No entanto, em vez de ser exclusivamente associada a uma pessoa, o propósito do perfil é ser assumido por qualquer pessoa que precisar dele.

O IAM concede aos usuários do IAM e ao usuário-raiz credenciais de longo prazo e credenciais temporárias dos perfis do IAM. usuários no Centro de Identidade do AWS IAM, entidades principais federadas OIDC ou SAML assumem perfis do IAM quando fazem login na AWS, que concede a eles credenciais temporárias. Como prática recomendada, aconselhamos que você exija que usuários humanos e workloads acessem recursos da AWS usando credenciais temporárias.

Diferenças entre usuários do IAM e usuários do IAM Identity Center

Os usuários do IAM não são contas distintas: são usuários dentro da sua conta. Cada usuário tem sua própria senha de acesso ao Console de gerenciamento da AWS. Você também pode criar uma chave de acesso individual para cada usuário, para que ele possa fazer solicitações programáticas para trabalhar com recursos em sua conta.

Os usuários do IAM e suas chaves de acesso recebem credenciais de longo prazo para seus recursos da AWS. O principal uso para usuários do IAM é oferecer às workloads que não podem usar perfis do IAM a capacidade de fazer solicitações programáticas a serviços da AWS usando a API ou a CLI.

> **Note**
>
> Para cenários em que você precise de usuários do IAM com acesso programático e credenciais de longo prazo, recomendamos atualizar as chaves de acesso quando necessário. Para obter mais informações, consulte Atualizar chaves de acesso.

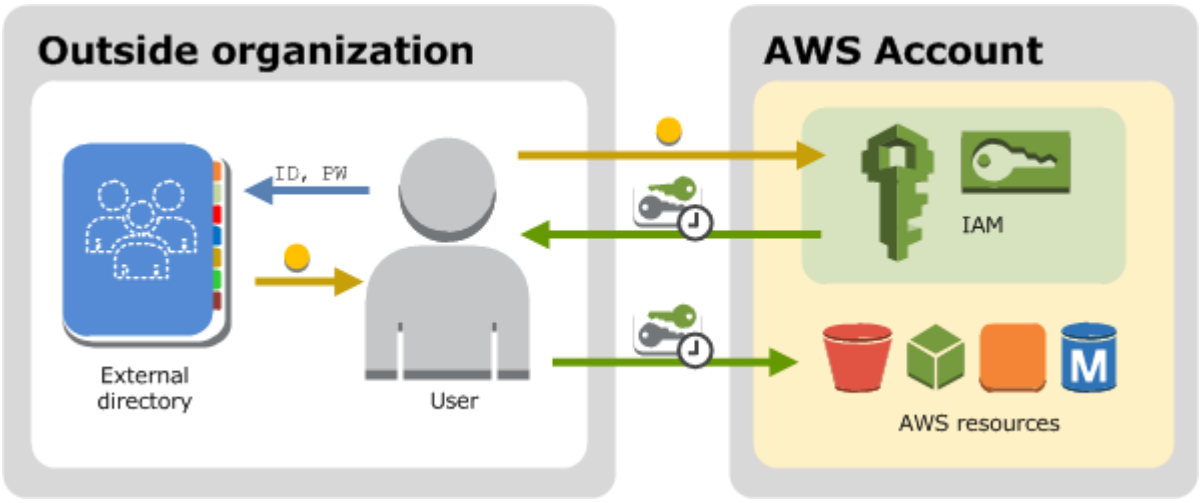
As identidades da força de trabalho (pessoas) são usuários no Centro de Identidade do AWS IAM que têm necessidades de permissões diferentes, dependendo da função que estão desempenhando, e podem trabalhar em várias Contas da AWS de uma organização. Se você tiver casos de uso que exijam chaves de acesso, poderá atender a esses casos de uso com o usuários no Centro de Identidade do AWS IAM. As pessoas que iniciam sessão via AWS podem obter chaves de acesso com credenciais de curto prazo para seus recursos da AWS. Para gerenciamento de acesso centralizado, recomendamos que você usar o Centro de Identidade do AWS IAM (IAM

Identity Center) para gerenciar o acesso às suas contas e as permissões nessas contas. O IAM Identity Center é configurado automaticamente com um diretório do Identity Center como fonte de identidades padrão, onde você pode criar usuários e grupos e atribuir o nível de acesso deles aos recursos da AWS. Para obter mais informações, consulte O que é o Centro de Identidade do AWS IAM no Guia do usuário do Centro de Identidade do AWS IAM.

A principal diferença entre esses dois tipos de usuário é que os usuários do IAM Identity Center assumem automaticamente um perfil do IAM quando entram no AWS antes de acessar o console de gerenciamento ou os recursos da AWS. Os perfis do IAM concedem credenciais temporárias toda vez que o usuário entra no AWS. Para os usuários do IAM entrarem usando um perfil do IAM, devem ter permissão para assumir e alternar perfis, e devem escolher explicitamente alternar para o perfil que desejarem assumir depois de acessar a conta da AWS.

Federar usuários a partir de uma fonte de identidades existente

Se os usuários da sua organização já tiverem sido autenticados quando entrarem na rede corporativa, você não precisará criar usuários do IAM ou usuários do IAM Identity Center separados para eles. Em vez disso, você pode criar a federação dessas identidades de usuários na AWS usando o IAM ou o Centro de Identidade do AWS IAM. As entidades principais federadas OIDC e SAML assumem um perfil do IAM que dá a elas permissões para acessar recursos específicos. Para ter mais informações sobre funções, consulte Termos e conceitos das funções.



A federação é útil nestes casos:

- Seus usuários já existem em um diretório corporativo.

  Se o seu diretório corporativo for compatível com Security Assertion Markup Language 2.0 (SAML 2.0), você poderá configurar o diretório corporativo para fornecer acesso de logon único (SSO) ao

Console de gerenciamento da AWS para seus usuários. Para obter mais informações, consulte Cenários comuns para credenciais temporárias.

Se o seu diretório corporativo não for compatível com SAML 2.0, você poderá criar uma aplicação de intermediador de identidades para fornecer aos usuários acesso com autenticação única (SSO) ao Console de gerenciamento da AWS. Para obter mais informações, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.

Se o seu diretório corporativo for o Microsoft Active Directory, você poderá usar o Centro de Identidade do AWS IAM para se conectar a um diretório autogerenciado no Active Directory ou um diretório no AWS Directory Service para estabelecer confiança entre o diretório corporativo e sua Conta da AWS.

Caso esteja usando um provedor de identidades (IdP) externo, como o Okta ou o Microsoft Entra, para gerenciar usuários, você poderá usar o Centro de Identidade do AWS IAM para estabelecer confiança entre seu IdP e sua Conta da AWS. Para obter mais informações, consulte Conectar-se a um provedor de identidades externo no Guia do usuário do Centro de Identidade do AWS IAM.

- Seus usuários já têm identidades da Internet.

  Se você estiver criando um aplicativo móvel ou um aplicativo baseado na web que permita aos usuários se identificar por meio de um provedor de identidade da Internet, como Login with Amazon, Facebook, Google ou qualquer provedor de identidade compatível com OpenID Connect (OIDC), o aplicativo poderá usar a federação para acessar a AWS. Para obter mais informações, consulte Federação OIDC.

  > **Tip**
  >
  > Para usar a federação de identidades com provedores de identidade da Internet, recomendamos usar o Amazon Cognito.

Diferentes métodos de fornecer acesso aos usuários

Veja a seguir como você pode fornecer o acesso a seus recursos da AWS.

Tipo de acesso do usuário	Quando é usado?	Onde encontrar mais informações?
Acesso aos recursos da AWS com autenticação única para pessoas, como usuários da força de trabalho, usando o IAM Identity Center	O Centro de Identidade do IAM fornece um local centralizado que reúne a administração de usuários e o acesso deles a Contas da AWS e aplicações em nuvem. É possível configurar um repositório de identidades no Centro de Identidade do IAM ou configurar a federação com um provedor de identidades (IdP) existente. As práticas recomendadas de segurança aconselham conceder aos usuários humanos credenciais limitadas para os recursos da AWS. As pessoas têm uma experiência de entrada mais fácil e você mantém controle sobre o acesso delas aos recursos em um único sistema. O Centro de Identidade do IAM oferece suporte a autenticação multifator (MFA) para aumentar a segurança da conta.	Para obter informações sobre como configurar o Centro de Identidade do IAM, consulte Getting Started (Conceitos básicos) no Guia do usuário do Centro de Identidade do AWS IAM Para obter mais informações sobre a MFA no Centro de Identidade do IAM, consulte Multi-factor authentication (Autenticação multifator) no Guia do usuário do Centro de Identidade do AWS IAM
Acesso federado	O IAM oferece suporte a IdPs compatíveis com OpenID	Para obter mais informações sobre provedores de identidade do IAM e federação, consulte

Tipo de acesso do usuário	Quando é usado?	Onde encontrar mais informações?
para usuários humanos, como usuários da força de trabalho, aos serviços da AWS que usam provedores de identidades (IdP) do IAM	Connect (OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Depois de criar um provedor de identidades do IAM, crie um ou mais perfis do IAM que podem ser atribuídos dinamicamente a uma entidade principal federada.	Federação e provedores de identidade na AWS.
Acesso entre contas para Contas da AWS	Você deseja compartilhar o acesso a determinados recursos da AWS com usuários em outras Contas da AWS. Os perfis são a principal forma de conceder acesso entre contas. Porém, alguns serviços da AWS são compatíveis com políticas baseadas em recurso que permitem anexar uma política diretamente a um recurso (em vez de usar um perfil como proxy).	Para obter mais informações sobre perfis do IAM, consulte Perfis do IAM. Para obter mais informações sobre funções vinculadas ao serviço, consulte Criar um perfil vinculado ao serviço. Para obter informações sobre quais serviços dão suporte ao uso de perfis vinculados a serviços, consulte AWSServiços da que funcionam com o IAM. Localize os serviços que têm Sim na coluna Perfil vinculado ao serviço. Para visualizar a documentação do perfil vinculado ao serviço desse serviço, selecione o link associado a Sim na coluna.

Tipo de acesso do usuário	Quando é usado?	Onde encontrar mais informações?
Credenciais de longo prazo para usuários do IAM designados na sua Conta da AWS	Poderá haver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM na AWS. Use o IAM para criar esses usuários do IAM na sua Conta da AWS e utilize o IAM para gerenciar suas permissões. Alguns casos de uso incluem: • Workloads que não podem usar perfis do IAM • Clientes da AWS de terceiros que exigem acesso programático por meio de chaves de acesso • Credenciais específicas de serviço para o AWS CodeCommit ou Amazon Keyspaces • Centro de Identidade do AWS IAMO não está disponível para sua conta, e você não tem outro provedor de identidade Como prática recomendada, para cenários em que você precisa de usuários do IAM com acesso programático e credenciais de longo prazo, recomendamos atualizar as	Para obter mais informações sobre como configurar um usuário do IAM, consulte Criar um usuário do IAM na Conta da AWS. Para obter mais informações sobre chaves de acesso do usuário do IAM, consulte Gerenciar chaves de acesso para usuários do IAM. Para obter mais informações sobre credenciais específicas de serviço para o AWS CodeCommit ou Amazon Keyspaces, consulte Credenciais do IAM para o CodeCommit: credenciais do Git, chaves SSH e chaves de acesso da AWS e Usar o IAM com o Amazon Keyspaces (para Apache Cassandra).

Tipo de acesso do usuário	Quando é usado?	Onde encontrar mais informações?
	chaves de acesso quando necessário. Para obter mais informações, consulte Atualizar chaves de acesso.	

Compatibilidade com acesso programático dos usuários

Os usuários precisarão de acesso programático se quiserem interagir com a AWS fora do Console de gerenciamento da AWS. A forma de conceder acesso programático depende do tipo de usuário que está acessando a AWS:

- Se você gerencia identidades no Centro de Identidade do IAM, as APIs da AWS exigem um perfil e a AWS Command Line Interface exige um perfil ou uma variável de ambiente.
- Se você tiver usuários do IAM, as APIs da AWS e a AWS Command Line Interface exigem chaves de acesso. Sempre que possível, crie credenciais temporárias, que consistem em um ID de acesso, uma chave de acesso secreta e um token de segurança que indica quando as credenciais expiram.

Para conceder acesso programático aos usuários, selecione uma das seguintes opções:

Qual usuário precisa de acesso programático?	Opção	Mais informações
Identidades da força de trabalho (Pessoas e usuários gerenciados no IAM Identity Center)	Use credenciais de curto prazo para assinar solicitaç ões programáticas para a AWS CLI ou as APIs da AWS (diretamente ou usando os SDKs da AWS).	Para a AWS CLI, siga as instruções em Como obter credenciais de perfil do IAM para acesso à CLI no Guia do usuário do Centro de Identidade do AWS IAM. Para as APIs da AWS, siga as instruções em Credenciais do

Qual usuário precisa de acesso programático?	Opção	Mais informações
		SSO no Guia de referência de AWS SDKs e ferramentas.
Usuários do IAM	Use credenciais de curto prazo para assinar solicitaç ões programáticas para a AWS CLI ou as APIs da AWS (diretamente ou usando os SDKs da AWS).	Siga as instruções em Usar credenciais temporárias com recursos da AWS.
Usuários do IAM	Use credenciais de longo prazo para assinar solicitaç ões programáticas para as APIs da AWS ou a AWS CLI (diretamente ou usando os AWS SDKs). (Não recomendado)	Siga as instruções em Gerenciamento de chaves de acesso de usuários do IAM.
Entidades principais federadas	Use uma operação de API do AWS STS para criar uma nova sessão com credenciais de segurança temporárias que incluem um par de chaves de acesso e um token de sessão.	Para obter explicações sobre as operações de API, consulte the section called “Solicita r credenciais de segurança temporárias”

Como permissões e políticas fornecem gerenciamento de acesso

A parte de gerenciamento de acesso do AWS Identity and Access Management (IAM) ajuda você a definir o que uma entidade principal pode fazer em uma conta. Uma entidade principal é uma pessoa ou aplicação autenticada usando uma entidade do IAM (usuário ou perfil do IAM). O gerenciamento de acesso geralmente é referenciado como autorização. Você gerencia o acesso na AWS criando políticas e anexando-as a identidades de IAM (usuários do IAM, grupos do IAM ou perfis IAM) ou recursos da AWS. Uma política é um objeto no AWS que, quando associado a uma identidade ou

recurso, define suas permissões. A AWS avalia essas políticas quando uma entidade de segurança usa uma entidade (usuário ou perfil) do IAM para fazer uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada na AWS como documentos JSON. Para obter mais informações sobre os tipos e os usos de políticas, consulte Políticas e permissões no AWS Identity and Access Management.

Políticas e contas

Se você gerenciar uma única conta na AWS, você definirá as permissões nessa conta usando políticas. Se você gerenciar permissões entre várias contas, será mais difícil gerenciar as permissões de seus usuários do IAM. Você pode usar funções do IAM, políticas baseadas em recurso ou listas de controle de acesso (ACLs) para obter permissões entre contas. No entanto, se você possuir várias contas, em vez disso, recomendamos o uso do serviço AWS Organizations para ajudá-lo a gerenciar essas permissões. Para obter mais informações, consulte O que é o AWS Organizations? no Guia do usuário do AWS Organizations.

Políticas e usuários

Usuários do IAM são identidades no Conta da AWS. Quando você cria um usuário do IAM, ele não pode acessar nada em sua conta até que você conceda permissão a ele. Você concede permissões a um usuário do IAM criando uma política baseada em identidade que é uma política anexada ao usuário do IAM ou a um grupo do IAM ao qual o usuário do IAM pertence. O exemplo a seguir mostra uma política JSON que permite que o usuário do IAM execute todas as ações do Amazon DynamoDB (`dynamodb:*`) na tabela `Books` na conta `123456789012` na região `us-east-2`.

JSON

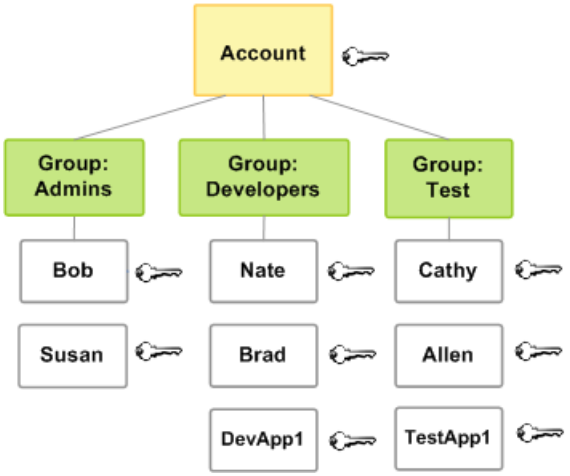
```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Depois que você anexa essa política ao usuário do IAM, ele tem permissão de realizar todas as ações da tabela `Books` da instância do DynamoDB. A maioria dos usuários do IAM tem várias políticas que são combinadas para representar todas as permissões concedidas a ele.

Por padrão, as ações ou recursos que não são explicitamente permitidos são negados. Por exemplo, se a política precedente for a única política anexada a um usuário, esse usuário poderá realizar as ações do DynamoDB na tabela `Books`, mas não poderá realizar ações em outras tabelas. De modo similar, o usuário não poderá realizar nenhuma ação no Amazon EC2, no Amazon S3 nem em qualquer outro serviço da AWS, porque as permissões para trabalhar com esses serviços não estão incluídas na política.

Políticas e grupos do IAM

Você pode organizar os usuários do IAM em grupos do IAM e anexar uma política a um grupo do IAM. Nesse caso, os usuários individuais do IAM ainda têm suas próprias credenciais, mas todos os usuários do IAM no grupo do IAM têm as permissões anexadas ao grupo. Use grupos do IAM para facilitar o gerenciamento de permissões.



Usuários ou grupos do IAM podem ter várias políticas anexadas a eles que concedem diferentes permissões. Nesse caso, a combinação de políticas determina as permissões efetivas da entidade principal. Se a entidade principal não tiver a permissão `Allow` explícita para uma ação e um recurso, a entidade principal não terá essas permissões.

Sessões de usuários federados e perfis

As entidades principais federadas não têm identidades permanentes na sua Conta da AWS da mesma forma que os usuários do IAM. Para atribuir permissões a entidades principais federadas,

é possível criar uma entidade denominada perfil e definir permissões para o perfil. Quando uma entidade principal federada SAML ou OIDC fizer login na AWS, o usuário será associado ao perfil e receberá as permissões definidas no perfil. Para obter mais informações, consulte Criar um perfil para um provedor de identidade de terceiros.

Políticas baseadas em identidade e em recursos

As políticas baseadas em identidade são políticas de permissões que você anexa a uma identidade do IAM, como um usuário, grupo ou função do IAM. As políticas baseadas em recurso são políticas de permissões que você anexa a um recurso, como um bucket do Amazon S3 ou uma política de confiança de função do IAM.

As Políticas baseadas em identidade controlam quais ações cada identidade pode realizar, em quais recursos e em que condições. As políticas baseadas em identidade podem ser categorizadas em:

- Políticas gerenciadas: políticas autônomas baseadas em identidade que você pode anexar a vários usuários, grupos e perfis na sua Conta da AWS. Você pode usar dois tipos de políticas gerenciadas:
  - Políticas gerenciadas pela AWS: políticas gerenciadas que são criadas e gerenciadas pela AWS. Se você não tiver experiência com o uso de políticas, recomendamos começar usando políticas gerenciadas pela AWS.
  - Políticas gerenciadas pelo cliente: políticas gerenciadas que você criar e gerenciar na sua Conta da AWS. As políticas gerenciadas pelo cliente oferecem um controle mais preciso de suas políticas do que as políticas gerenciadas pela AWS. Você pode criar, editar e validar uma política do IAM no editor visual ou criando o documento de política JSON diretamente. Para obter mais informações, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente e Editar políticas do IAM.
- Políticas em linha: políticas que você cria e gerencia e que são incorporadas diretamente em um único usuário, grupo ou função. Na maioria dos casos, não recomendamos o uso de políticas embutidas.

As Políticas baseadas em recurso controlam quais ações uma entidade principal pode realizar nesses recursos, e em que condições. As políticas baseadas em recursos são políticas em linha, e não há políticas gerenciadas que sejam baseadas em recurso. Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em atributo.

O serviço do IAM é compatível com um único tipo de política baseada em recurso denominada política de confiança de um perfil, a qual você anexa a um perfil do IAM. Como um perfil do IAM é tanto uma identidade quanto um recurso que é compatível com as políticas baseadas em recurso, você deve anexar uma política de confiança e uma política baseada em identidade a um perfil do IAM. As políticas de confiança definem quais entidades principais (contas, usuários, perfis e entidades principais de usuários federados do AWS STS) podem assumir o perfil. Para saber como as funções do IAM diferem de outras políticas baseadas em recurso, consulte Acesso a recursos entre contas no IAM.

Para ver quais serviços oferecem suporte a políticas baseadas em recursos, consulte AWSServiços da que funcionam com o IAM. Para saber mais sobre as políticas baseadas em recursos, consulte Políticas baseadas em identidade e em recurso.

Definir permissões com base em atributos com autorização ABAC

Controle de acesso por atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos. A AWS chama esses atributos de tags. Você pode anexar tags a recursos do IAM, incluindo entidades do IAM (usuários ou perfis do IAM) e a recursos da AWS. É possível criar uma única política de ABAC ou um pequeno conjunto de políticas para suas entidades do IAM. Você pode criar políticas de ABAC que permitem operações quando a tag da entidade principal corresponde à tag de recurso. O sistema de atributos do ABAC que fornece tanto contexto de usuário de nível alto quanto controle de acesso granular. Como o ABAC é baseado em atributo, ele pode realizar a autorização dinâmica de dados ou aplicações, concedendo ou revogando acesso em tempo real. O ABAC é útil em ambientes que estão escalando e em situações em que o gerenciamento de políticas de identidades ou de recursos se tornou um problema.

Por exemplo, é possível criar três perfis do IAM com a chave de tag `access-project`. Defina o valor da tag do primeiro perfil do IAM como `Heart`, o segundo como `Star` e o terceiro como `Lightning`. Você então pode usar uma única política que permita acesso quando o perfil do IAM e o recurso da AWS tiverem uma tag `access-project`. Para obter um tutorial detalhado que demonstra como usar o ABAC na AWS, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas. Para saber mais sobre os serviços compatíveis com ABAC, consulte AWSServiços da que funcionam com o IAM.
Esse diagrama mostra que as tags aplicadas a uma entidade principal devem corresponder às tags aplicadas a um recurso para que o usuário tenha permissões de acesso ao recurso. As tags devem ser aplicadas a grupos de recursos, usuários individuais, recursos individuais e grupos do IAM.

Comparação do ABAC com o modelo RBAC tradicional

O modelo de autorização tradicional usado no IAM é o controle de acesso por perfil (RBAC). O RBAC define permissões com base no cargo ou no perfil de uma pessoa, o que é diferente de um perfil do IAM. O IAM inclui políticas gerenciadas para funções de trabalho que alinham as permissões para uma função de trabalho em um modelo de RBAC.

No IAM, você implementa o RBAC criando diferentes políticas para diferentes funções de trabalho. Em seguida, você anexa as políticas às identidades (usuários, grupos ou perfis do IAM). Como prática recomendada, conceda as permissões mínimas necessárias para o perfil de trabalho. Isso resulta em acesso com privilégio mínimo. Cada política de trabalho lista os recursos específicos que as identidades que têm essa política atribuída a elas podem acessar. A desvantagem de usar o modelo RBAC tradicional é que, quando você ou seus usuários adicionam novos recursos ao ambiente, é necessário atualizar as políticas para permitir o acesso a esses recursos.

Por exemplo, vamos supor que você tenha três projetos, chamados `Heart`, `Star` e `Lightning`, nos quais seus funcionários trabalham. Você cria uma função do IAM para cada projeto. Depois, você anexa políticas a cada perfil do IAM para definir os recursos que qualquer pessoa com permissão para assumir esse perfil pode acessar. Se um funcionário mudar de cargo na sua empresa, você atribuirá a ele outra função do IAM. Você pode atribuir mais de um perfil do IAM a pessoas ou a programas. No entanto, o projeto `Star` pode exigir recursos adicionais, como um novo contêiner do Amazon EC2. Nesse caso, é necessário atualizar a política anexada ao perfil do IAM `Star` para especificar o novo recurso de contêiner. Caso contrário, os membros do projeto `Star` não terão permissão para acessar o novo contêiner.
Esse diagrama ilustra que o controle de acesso baseado em perfil exige que cada identidade seja atribuída a uma política baseada em cargo específica para acessar recursos diferentes.

O ABAC oferece as seguintes vantagens em relação ao modelo de RBAC tradicional:

- As permissões de ABAC são dimensionadas com inovação. Não é mais necessário que um administrador atualize as políticas existentes para permitir o acesso a novos recursos. Por exemplo, vamos supor que você tenha criado sua estratégia de ABAC com a tag `access-project`. Um desenvolvedor usa o perfil do IAM com a tag `access-project` = `Heart`. Quando as pessoas no projeto `Heart` precisarem de recursos adicionais do Amazon EC2, o desenvolvedor poderá criar novas instâncias do Amazon EC2 com a etiqueta `access-project` = `Heart`. Depois disso, qualquer pessoa no projeto `Heart` pode iniciar e interromper essas instâncias porque seus valores de tag são correspondentes.

- O ABAC exige menos políticas. Como não é necessário criar políticas diferentes para funções de trabalho diferentes, você cria menos políticas. Essas políticas são mais fáceis de gerenciar.
- Usando o ABAC, as equipes podem responder de modo dinâmico a mudança e crescimento. Como as permissões para novos recursos são concedidas automaticamente com base em atributos, você não precisa atribuir políticas às identidades manualmente. Por exemplo, se a empresa já oferece suporte aos projetos `Heart` e `Star` que usam o ABAC, é fácil adicionar um novo projeto `Lightning`. Um administrador do IAM cria um novo perfil do IAM com a tag `access-project` = `Lightning`. Não é necessário alterar a política para oferecer suporte a um novo projeto. Qualquer pessoa com permissões para assumir o perfil do IAM pode criar e visualizar instâncias marcadas com `access-project` = `Lightning`. Outro cenário é quando um membro da equipe muda do projeto `Heart` para o projeto `Lightning`. Para fornecer aos membros da equipe acesso ao projeto `Lightning`, o administrador do IAM atribui a eles outro perfil do IAM. Não é necessário alterar as políticas de permissões.
- Permissões granulares são possíveis usando ABAC. Ao criar políticas, faz parte das melhores práticas conceder privilégio mínimo. Usando o RBAC tradicional, você escreve uma política que permite o acesso a recursos específicos. Porém, quando você usa o ABAC, pode permitir ações em todos os recursos se a tag de recurso corresponder à tag da entidade principal.
- Use atributos de funcionário do seu diretório corporativo com ABAC. Você pode configurar seu provedor SAML ou OIDC para passar tags de sessão para o IAM. Quando seus funcionários formam uma federação na AWS, o IAM aplica os atributos desses funcionários à entidade principal. Você pode usar o ABAC para conceder ou não permissões com base nesses atributos.

Para obter um tutorial detalhado que demonstra como usar o ABAC na AWS, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas.

Conceitos básicos do IAM

O AWS Identity and Access Management (IAM) ajuda você a controlar com segurança o acesso à Amazon Web Services (AWS) e aos recursos de sua conta. O IAM também pode manter privadas as credenciais de login. Você não precisa se cadastrar especificamente para usar o IAM. Não há custo pelo uso do IAM.

A forma como você usa o IAM depende da sua função de trabalho. Consulte Quando posso usar o IAM? para obter orientações específicas para cada perfil.

Use o IAM para fornecer identidades, como usuários e perfis, e acesso a recursos de sua conta. Por exemplo, você pode usar o IAM com usuários existentes no diretório corporativo que você gerencia externamente para a AWS ou criar usuários na AWS usando o Centro de Identidade do AWS IAM. As identidades federadas assumem perfis do IAM definidos para acessar os recursos necessários. Para obter mais informações sobre o IAM Identity Center, consulte What is IAM Identity Center? (O que é o IAM Identity Center?) no Guia do usuário do Centro de Identidade do AWS IAM.

> **Note**
>
> O IAM é integrado a vários produtos da AWS. Para obter uma lista de serviços compatíveis com o IAM, consulte AWSServiços da que funcionam com o IAM.

Para saber como começar com a AWS, criar um usuário administrativo, um AWS Organizations e usar vários serviços para resolver um problema, como criar e lançar seu primeiro projeto, consulte o Centro de recursos de conceitos básicos.

Para praticar com tarefas comuns do IAM, consulte Tutoriais do IAM. As tarefas incluem a criação de perfis para acesso entre contas e o uso de tags para controle de acesso por atributo.

Depois de configurar sua conta, planeje como as pessoas e as workloads a acessam. Para obter mais informações, consulte Planeje o acesso à sua conta da AWS.

Em seguida, use Preparar para permissões de privilégio mínimo para seguir o princípio do privilégio mínimo.

Configurar a Conta da AWS

Antes de começar a trabalhar com o IAM, certifique-se de que concluiu a configuração inicial do seu ambiente da AWS.

AWSA envia um e-mail de confirmação depois que o processo de inscrição é concluído. A qualquer momento, é possível exibir as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Quando se cadastrou no serviço, você criou uma Conta da AWS usando um endereço de e-mail e uma senha. Estas são as suas credenciais de usuário raiz da AWS. Como prática recomendada, você não deve usar as credenciais de usuário-raiz para acessar a AWS e realizar tarefas diárias. Só use suas credenciais de usuário-raiz para realizar tarefas que exijam as credenciais de usuário-raiz. Além disso, não compartilhe suas credenciais com ninguém. Em vez disso, adicione as pessoas ao seu diretório e dê a elas acesso à sua Conta da AWS.

Para proteger seu Usuário raiz da conta da AWS

1. Faça login no Console de gerenciamento da AWS como o proprietário da conta ao escolher a opção Usuário-raiz e inserir o endereço de e-mail da Conta da AWS. Na próxima página, insira a senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS.
2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo MFA virtual para sua Conta da AWS de usuário-raiz (console) no Guia do usuário do IAM.

Conceder acesso ao console de faturamento

Por padrão, os usuários e perfis do IAM em uma Conta da AWS não têm acesso ao console do Gerenciamento de Faturamento e Custos. Isso acontece mesmo que eles tenham políticas do IAM que concedam acesso a determinados atributos de Faturamento. Para conceder acesso, o usuário-raiz da Conta da AWS deve primeiro ativar o acesso ao IAM.

> **Note**
>
> Como prática recomendada de segurança, recomendamos que você ofereça acesso aos seus recursos por meio da federação de identidades, com o Centro de Identidade do AWS IAM. Quando você habilita o Centro de Identidade do IAM em conjunto com o AWS Organizations, o console do Gerenciamento de Faturamento e Custos é habilitado por padrão com o faturamento consolidado para todas as Contas da AWS de sua organização. Para obter mais informações, consulte Consolidating billing for AWS Organizations no Guia do usuário do Gerenciamento de Faturamento e Custos.

1. Faça login no Console de gerenciamento da AWS com suas credenciais de usuário raiz (especificamente, o endereço de e-mail e a senha usados para criar sua AWS).
2. Na barra de navegação, escolha o nome da conta e, depois, escolha Minha conta.
3. Role a página para baixo até encontrar a seção Acesso do usuário e do perfil do IAM a informações de faturamento e selecione Editar.
4. Marque a caixa de seleção Ativar acesso ao IAM para ativar o acesso às páginas do console do Gerenciamento de Faturamento e Custos.
5. Selecione Atualizar.

   A página exibe a mensagem de que o acesso do usuário/perfil do IAM às informações de faturamento está ativado.

   > **Important**
   >
   > Ativar o acesso ao IAM isoladamente não concede aos usuários e perfis do IAM as permissões para visualizar as páginas do console do Gerenciamento de Faturamento e Custos. Você também deve anexar as políticas baseadas em identidade necessárias aos perfis do IAM para conceder acesso ao console de faturamento. Os perfis fornecem credenciais temporárias que os usuários podem assumir quando necessário.

6. Use o Console de gerenciamento da AWS para criar um perfil que um usuário possa assumir para acessar o console de faturamento.
7. Na página Adicionar permissões para o perfil, adicione permissões para listar e visualizar detalhes sobre os recursos de faturamento na sua Conta da AWS.

A política gerenciada de Faturamento da AWS concede aos usuários permissão para visualizar e editar o console do Gerenciamento de Faturamento e Custos. Isso inclui visualizar o uso da conta, modificar orçamentos e métodos de pagamento. Para ver mais exemplos de políticas que você pode anexar aos perfis do IAM para controlar o acesso às informações de faturamento da sua conta, consulte AWS Billing policy examples no Guia do usuário do Gerenciamento de Faturamento e Custos.

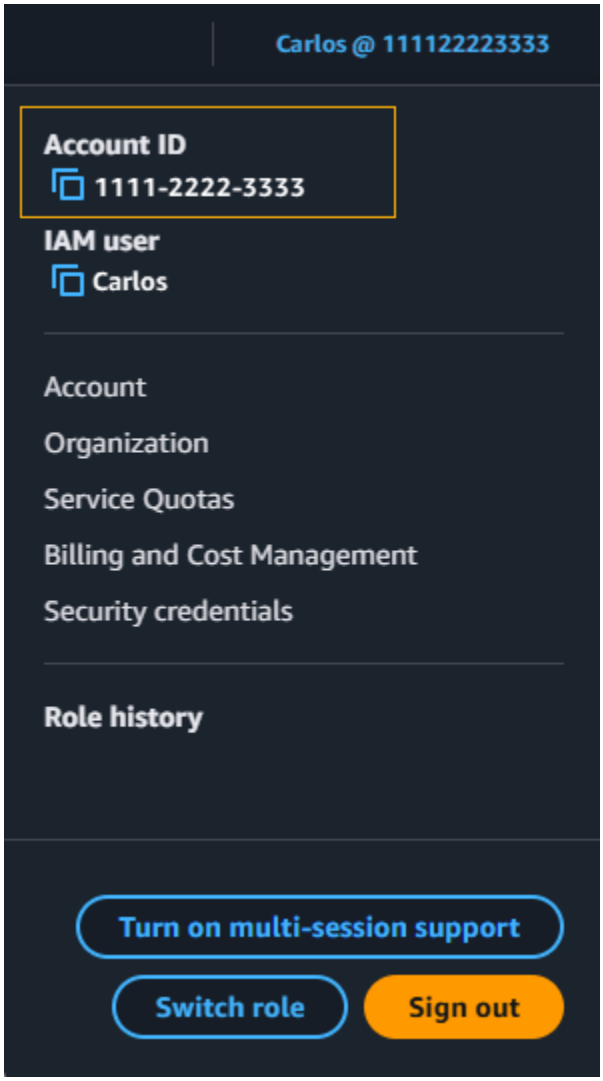
Visualizar o ID da Conta da AWS

Caso esteja conectado ao console, você pode visualizar o ID da conta da Conta da AWS usando os métodos a seguir.

Para visualizar sua ID da Conta da AWS

Console

O ID da conta da AWS é exibido quando você vai para o Painel do IAM na seção Conta da AWS. Você também pode ver o ID da sua conta na barra de navegação no canto superior direito. Escolha seu nome de usuário e o ID da conta será exibido acima do seu nome de usuário.



AWS CLI

Use o comando a seguir para visualizar o ID do usuário, o ID da conta e o ARN do usuário:

- aws sts get-caller-identity

API

Use a API a seguir para visualizar o ID do usuário, o ID da conta e o ARN do usuário:

- GetCallerIdentity

Usar um alias para seu ID da Conta da AWS

O ID da sua conta da é um número de 12 dígitos que identifica sua conta de maneira exclusiva. Por padrão, os usuários do IAM na conta fazem login usando um URL da web que inclui o ID da conta. Se não tiverem a URL, ao entrar, eles poderão fornecer o ID da conta na página de entrada da AWS.

O URL de uma página de login tem o seguinte formato, por padrão.

```
https://Your_Account_ID.signin.aws.amazon.com/console/
```

Muitas pessoas acham mais fácil lembrar de palavras que de números; portanto, criar um alias para o ID da conta pode ajudar os usuários do IAM a entrar com mais facilidade.

Se você criar um alias da Conta da AWS para o ID da sua Conta da AWS, o URL da página de login terá a aparência do exemplo a seguir.

```
https://Your_Account_Alias.signin.aws.amazon.com/console/
```

Considerações antes de criar um alias de conta

- Sua Conta da AWS pode ter apenas um alias. Se você criar um novo alias para sua conta da AWS, o novo alias substituirá o anterior, e o URL que contém o alias anterior deixará de funcionar.
- O alias da conta deve conter apenas dígitos, letras minúsculas e hifens. Para obter mais informações sobre as limitações de entidades de contas da AWS, consulte IAM e cotas do AWS STS.
- O alias da conta deve ser exclusivo em todos os produtos da Amazon Web Services em uma determinada partição de rede.

  Uma partição é um grupo de regiões da AWS. Cada conta da AWS tem escopo para uma partição.

  Estas são as partições compatíveis:

  - `aws`: regiões da AWS
  - `aws-cn`: regiões da China
  - `aws-us-gov`: regiões da AWS GovCloud (US)

> **Note**
>
> Os aliases de conta não são segredos e aparecerão na URL da página de login visível ao público. Não inclua nenhuma informação confidencial no alias da conta.
> O URL original que contém o ID da sua Conta da AWS permanece ativo e poderá ser usado depois que você cria o alias da sua Conta da AWS.

Para obter instruções sobre como configurar ou remover um alias, consulte Criar um alias de conta e Excluir um alias de conta.

Criar um alias de conta

Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:

- `iam:ListAccountAliases`
- `iam:CreateAccountAlias`

Para criar um alias da Conta da AWS

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Painel.
3. Na seção Conta da AWS, ao lado de Alias da conta, escolha Criar. Se um alias já existir, escolha Edit (Editar).
4. Na caixa de diálogo, digite o nome que você deseja usar para o alias e, em seguida, escolha Salvar alterações.

AWS CLI

Execute o seguinte comando:

- `aws iam create-account-alias`

API

Para criar um alias para o URL da página de login do Console de gerenciamento da AWS, chame a seguinte operação:

- `CreateAccountAlias`

Excluir um alias de conta

Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:

- `iam:ListAccountAliases`
- `iam:DeleteAccountAlias`

Como excluir um alias da conta

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Painel.
3. Na sessão Conta da AWS, ao lado de Alias da conta, escolha Excluir.

AWS CLI

Para excluir um alias de ID da Conta da AWS, execute o seguinte comando:

- `aws iam delete-account-alias`

Para confirmar que o alias de conta foi excluído, tente exibir seu alias de ID da Conta da AWS, executando o seguinte comando:

- `aws iam list-account-aliases`

API

Para excluir um alias de ID da sua Conta da AWS, chame a seguinte operação:

- `DeleteAccountAlias`

Para confirmar que o alias de conta foi excluído, tente exibir seu alias de ID da Conta da AWS, chamando a seguinte operação:

- `ListAccountAliases`

> **Note**
>
> Após excluir o alias de conta, o único URL de login para sua conta é baseada no ID da conta. Qualquer tentativa de se conectar ao URL do alias falhará e não será redirecionada.

Planeje o acesso à sua conta da AWS

Ao configurar a AWS, planeje como você pretende que as pessoas acessem sua conta e recursos da AWS. Uma solução de gerenciamento de identidade bem projetada mantém sua conta segura.

Fontes de identidade

De acordo com as práticas recomendadas do IAM, é melhor que usuários humanos e workloads usem credenciais temporárias para acessar os recursos da AWS. Você concede credenciais temporárias por meio de perfis do IAM. Tanto os usuários federados no IAM quanto os usuários no Centro de Identidade do IAM (federados ou criados no diretório do Centro de Identidade do IAM) usam perfis do IAM para acessar recursos.

Antes de começar a usar a AWS, planeje como configurar suas identidades usando uma das seguintes abordagens:

- Habilite o Centro de Identidade do IAM com o AWS Organizations e adicione usuários no Centro de Identidade do IAM diretamente ao diretório organizacional.

  Para obter instruções sobre como adicionar usuários diretamente ao diretório organizacional do IAM Identity Center, consulte Adicionar usuários
- Federe seu provedor de identidade externo existente com o Centro de Identidade do IAM ou o IAM.

Para obter instruções sobre como federar um provedor de identidade externo no diretório organizacional do IAM Identity Center, consulte o Tutorial de conceitos básicos apropriado.

Gerenciamento de acesso

Identifique os recursos e serviços da AWS de que seus usuários precisam. Em seguida, defina as permissões e políticas de acesso para cada usuário, grupo ou perfil.

- Se você usa o IAM Identity Center, um provedor de identidade do IAM, os perfis do IAM e as políticas de permissões são criados em cada conta da AWS da organização. Esses perfis e permissões correspondem ao que você especifica ao atribuir pessoas ou grupos a aplicativos ou contas da AWS.

  Para obter mais informações, consulte Assign user access e Set up single sign-on access to your applications.
- Você pode usar o gerenciador de acesso à conta — um atributo do IAM que permite atribuir perfis do IAM existentes que você cria e gerencia para os usuários e grupos do IAM Identity Center. O gerenciador de acesso à conta fornece acesso ao conjunto completo de atributos de perfis do IAM. Você pode usá-lo junto com os conjuntos de permissões ou sozinho.
- Se você federar seu provedor de identidade diretamente com o IAM na sua Conta da AWS, deverá criar um perfil para seus usuários assumirem. O perfil precisa de uma política de confiança e uma política de permissões. A política de confiança declara quem pode assumir o perfil. A política de permissões indica quais ações e atributos da AWS o perfil tem ou não permissão para acessar.

  Para obter mais informações, consulte . Federação e provedores de identidade na AWS

Para obter instruções sobre como adicionar uma camada extra de segurança para login, consulte Usar autenticação multifator com suas identidades.

Casos de uso de usuários do IAM

Os usuários do IAM criados na sua Conta da AWS têm credenciais de longo prazo que você gerencia diretamente.

Quando se trata de gerenciar o acesso na AWS, os usuários do IAM geralmente não são a melhor escolha. Há alguns motivos principais pelos quais você deve evitar depender dos usuários do IAM na maioria dos seus casos de uso.

Primeiro, os usuários do IAM são projetados para contas individuais, então eles não escalam bem à medida que sua organização cresce. Gerenciar permissões e segurança para um grande número de usuários do IAM pode rapidamente se tornar um desafio.

Os usuários do IAM também não têm os recursos centralizados de visibilidade e auditoria que você obtém com outras soluções de gerenciamento de identidade da AWS. Isso pode fazer com que manter a segurança e a conformidade regulatória seja mais difícil.

Por fim, implementar as práticas recomendadas de segurança, como autenticação multifatorial, políticas de senha e separação de perfis, é muito mais fácil com abordagens de gerenciamento de identidade mais escaláveis.

Em vez de utilizar usuários do IAM, recomendamos o uso de soluções mais robustas, como o IAM Identity Center com o AWS Organizations, ou de identidades federadas fornecidas por provedores externos. Essas opções proporcionarão melhor controle, segurança e eficiência operacional à medida que seu ambiente da AWS cresce.

Sendo assim, recomendamos usar somente usuários do IAM para casos de uso não compatíveis com usuários federados.

A lista seguinte identifica casos de uso específicos que exigem credenciais de longo prazo com usuários do IAM na AWS. Você pode usar o IAM para criar esses usuários do IAM sob o controle da sua conta da AWS e usar o IAM para gerenciar as permissões deles.

- Acesso emergencial à sua conta da AWS
- Workloads que não podem usar perfis do IAM
  - AWS CodeCommitacesso do
  - Acesso ao Amazon Keyspaces (para Apache Cassandra)
- Clientes da AWS de terceiros
- Centro de Identidade do AWS IAMO não está disponível para sua conta, e você não tem outro provedor de identidade

Criar um acesso de emergência para um usuário do IAM

Um usuário do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas para uma única pessoa ou aplicação.

Ter um usuário do IAM para acesso de emergência é um dos motivos recomendados para criar um usuário do IAM. Assim, você possa acessar sua Conta da AWS caso seu provedor de identidade não esteja acessível.

> **Note**
>
> Como prática recomendada de segurança, recomendamos que você forneça acesso aos seus recursos por meio da federação de identidades, em vez de criar usuários do IAM. Para obter informações sobre situações específicas em que um usuário do IAM é necessário, consulte Quando criar um usuário do IAM (em vez de um perfil).

Para criar um acesso de emergência para um usuário do IAM

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:
>
> - `access-analyzer:ValidatePolicy`
> - `iam:AddUserToGroup`
> - `iam:AttachGroupPolicy`
> - `iam:CreateGroup`
> - `iam:CreateLoginProfile`
> - `iam:CreateUser`
> - `iam:GetAccountPasswordPolicy`
> - `iam:GetLoginProfile`
> - `iam:GetUser`
> - `iam:ListAttachedGroupPolicies`
> - `iam:ListAttachedUserPolicies`
> - `iam:ListGroupPolicies`
> - `iam:ListGroups`
> - `iam:ListGroupsForUser`
> - `iam:ListPolicies`
> - `iam:ListUserPolicies`

- `iam:ListUsers`

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione Usuários e escolha Criar usuário.

> **Note**
>
> Se o Centro de Identidade do IAM estiver habilitado, o Console de gerenciamento da AWS exibirá um lembrete de que é melhor gerenciar o acesso dos usuários no Centro de Identidade do IAM. Nesse procedimento, o usuário do IAM que você criar será usado especificamente apenas quando seu provedor de identidade não estiver disponível.

4. Na página Especificar detalhes do usuário, em Detalhes do usuário, em Nome de usuário, insira o nome do novo usuário. É o nome de login para a AWS. Neste exemplo, insira **`EmergencyAccess`**.

> **Note**
>
> Os nomes de usuário podem ser uma combinação de até 64 letras, dígitos e estes caracteres: adição (+), igual (=), vírgula (,), ponto (.), arroba (@), sublinhado (_) e hífen (-). Os nomes devem ser exclusivos dentro de uma conta. Eles não são diferenciados por letras maiúsculas e minúsculas. Por exemplo, você não pode criar dois usuários denominados TESTUSER e testuser. Quando o nome de usuário é usado em uma política ou como parte de um ARN, o nome diferencia maiúsculas de minúsculas. Quando é exibido para os clientes no console, por exemplo, como durante o processo de login, o nome de usuário não diferencia maiúsculas de minúsculas.

5. Marque a caixa de seleção ao lado de Fornecer acesso ao Console de gerenciamento da AWS: opcional e escolha Quero criar um usuário do IAM.

6. Em Senha do console, selecione Senha gerada automaticamente.
7. Desmarque a caixa de seleção ao lado de O usuário deverá criar uma nova senha no próximo login (recomendado). Como esse usuário do IAM serve para acesso emergencial, um administrador confiável retém a senha e a fornece somente quando necessário.
8. Na página Definir permissões, em Opções de permissões, selecione Adicionar usuário ao grupo. Em seguida, em Grupos de usuários, selecione Criar grupo.
9. Na página Criar grupo de usuários, em Nome do grupo de usuários, insira **`EmergencyAccessGroup`**. Em seguida, em Políticas de permissões, selecione AdministratorAccess.
10. Selecione Criar grupo de usuários para voltar à página Definir permissões.
11. Em Grupos de usuários, selecione o nome do **`EmergencyAccessGroup`** que você criou anteriormente.
12. Selecione Avançar para prosseguir para a página Revisar e criar.
13. Na página Revisar e criar, revise a lista de associações de grupos de usuários a serem adicionadas ao novo usuário. Quando você estiver pronto para continuar, selecione Criar usuário.
14. Na página Recuperar senha, selecione Baixar arquivo .csv para salvar um arquivo .csv com as informações de credencial do usuário (URL da conexão, nome de usuário e senha).
15. Salve esse arquivo para usar caso precise fazer login no IAM e não tenha acesso ao provedor de identidade.

O novo usuário do IAM será exibido na lista Usuários. Selecione o link Nome de usuário para ver os detalhes do usuário.

AWS CLI

1. Crie um usuário chamado **`EmergencyAccess`**.

   - aws iam create-user

```
aws iam create-user \
   --user-name EmergencyAccess
```

2. (Opcional) Forneça ao usuário acesso ao Console de gerenciamento da AWS. Isso requer uma senha. Para criar uma senha de um usuário do IAM, você pode usar o parâmetro `--`

`cli-input-json` para transmitir um arquivo JSON que contém a senha. Você também deve fornecer ao usuário o URL da página de login da sua conta.

- aws iam create-login-profile

```
aws iam create-login-profile \
   --generate-cli-skeleton > create-login-profile.json
```

- Abra o arquivo `create-login-profile.json` em um editor de texto e digite uma senha que esteja em conformidade com sua política de senha. Depois, salve o arquivo. Por exemplo:

```
{
 "UserName": "EmergencyAccess",
 "Password": "Ex@3dRA0djs",
 "PasswordResetRequired": false
}
```

- Use o comando `aws iam create-login-profile` novamente, passando o parâmetro `--cli-input-json` para especificar seu arquivo JSON.

```
aws iam create-login-profile \
   --cli-input-json file://create-login-profile.json
```

> **Note**
>
> Se a senha que você forneceu no arquivo JSON violar a política de senha da sua conta, você receberá um erro de `PasswordPolicyViolation`. Se isso acontecer, revise a política de senha da sua conta e atualize a senha no arquivo JSON para atender aos requisitos.

3. Crie o **`EmergencyAccessGroup`**, anexe a política `AdministratorAccess` gerenciada pela AWS ao grupo e adicione o usuário **`EmergencyAccess`** ao grupo.

> **Note**
>
> Uma política gerenciada pela AWS é uma política independente que é criada e administrada pela AWS. Cada política tem seu próprio nome do recurso da Amazon (ARN) que inclui o nome da política. Por exemplo, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` é uma política gerenciada da AWS. Para obter mais informações sobre ARNs do , consulte ARNs do IAM. Para obter uma lista das políticas gerenciadas pela AWS dos Serviços da AWS, consulte Políticas gerenciadas pela AWS.

- aws iam create-group

```
aws iam create-group \
   --group-name EmergencyAccessGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
   --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
   --group-name >EmergencyAccessGroup
```

- aws iam add-user-to-group

```
aws iam add-user-to-group \
   --user-name EmergencyAccess \
   --group-name EmergencyAccessGroup
```

- Execute o comando aws iam get-group para listar o **EmergencyAccessGroup** e seus membros.

```
aws iam get-group \
   --group-name EmergencyAccessGroup
```

Crie um usuário do IAM para workloads que não podem usar perfis do IAM

> **Important**
>
> Como prática recomendada, recomendamos que você exija que os usuários humanos usem credenciais temporárias ao acessar a AWS.
> Como alternativa, você pode gerenciar as identidades de usuário, incluindo o usuário administrativo, com o Centro de Identidade do AWS IAM. Recomendamos usar o Centro de Identidade do IAM para gerenciar o acesso às contas e as permissões nelas. Se estiver usando um provedor de identidade externo, você também poderá configurar as permissões de acesso para identidades de usuário no Centro de Identidade do IAM.

Se o seu caso de uso exigir usuários do IAM com acesso programático e credenciais de longo prazo, recomendamos que você estabeleça procedimentos para atualizar as chaves de acesso quando necessário. Para obter mais informações, consulte Atualizar chaves de acesso.

Para executar algumas tarefas de gerenciamento de contas e serviços, é necessário fazer login usando as credenciais de usuário-raiz. Para visualizar as tarefas que exigem que você faça login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz.

Para criar um usuário do IAM para workloads que não podem usar perfis do IAM

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:
>
> - `iam:AddUserToGroup`
> - `iam:AttachGroupPolicy`
> - `iam:CreateAccessKey`
> - `iam:CreateGroup`
> - `iam:CreateServiceSpecificCredential`
> - `iam:CreateUser`
> - `iam:GetAccessKeyLastUsed`
> - `iam:GetAccountPasswordPolicy`
> - `iam:GetAccountSummary`
> - `iam:GetGroup`

- iam:GetLoginProfile
- iam:GetPolicy
- iam:GetRole
- iam:GetUser
- iam:ListAccessKeys
- iam:ListAttachedGroupPolicies
- iam:ListAttachedUserPolicies
- iam:ListGroupPolicies
- iam:ListGroups
- iam:ListGroupsForUser
- iam:ListInstanceProfilesForRole
- iam:ListMFADevices
- iam:ListPolicies
- iam:ListRoles
- iam:ListRoleTags
- iam:ListSSHPublicKeys
- iam:ListServiceSpecificCredentials
- iam:ListSigningCertificates
- iam:ListUserPolicies
- iam:ListUserTags
- iam:ListUsers
- iam:UploadSSHPublicKey
- iam:UploadSigningCertificate

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione Usuários e, em seguida, escolha Criar usuário.

4. Na página Especificar detalhes do usuário, faça o seguinte:

   a. Em Nome de usuário, digite ***`WorkloadName`***. Substitua ***`WorkloadName`*** pelo nome da workload que usará a conta.

   b. Escolha Próximo.

5. (Opcional) Na página Definir permissões, faça o seguinte:

   a. Escolha Add user to group.

   b. Escolha Criar grupo.

   c. Na caixa de diálogo Criar grupo de usuários, em Nome do grupo de usuários, digite um nome que represente o uso das workloads no grupo. Para este exemplo, use o nome **`Automation`**.

   d. Em Políticas de permissões, marque a caixa de seleção da política gerenciada PowerUserAccess.

   > **Tip**
   >
   > Insira Power na caixa de pesquisa Políticas de permissões para encontrar rapidamente a política gerenciada.

   e. Escolha Criar grupo de usuários.

   f. De volta à página com a lista de grupos do IAM, marque a caixa de seleção do novo grupo de usuários. Escolha Refresh (Atualizar) se não vir o novo grupo de usuários na lista.

   g. Escolha Próximo.

6. (Opcional) Na seção Tags, adicione metadados ao usuário associando tags como pares de chave/valor. Para obter mais informações, consulte Tags para recursos do AWS Identity and Access Management.

7. Verifique as associações de grupos de usuários para o novo usuário. Quando você estiver pronto para continuar, escolha Create user (Criar usuário).

8. É exibida uma notificação de status informando que o usuário foi criado com êxito. Selecione Visualizar usuário para acessar a página de detalhes do usuário

9. Selecione a guia Credenciais de segurança. Em seguida, crie as credenciais necessárias para a workload.

- Chaves de acesso: selecione Criar chave de acesso para gerar e baixar chaves de acesso para o usuário.

  > **Important**
  >
  > Esta é a única oportunidade de visualizar ou fazer download das chaves de acesso secretas, e você deve fornecer essas informações aos usuários para que eles possam usar a AWS API. Salve a nova ID da chave de acesso do usuário e a chave de acesso secreta em um local seguro e protegido. Você não terá acesso às chaves secretas novamente depois dessa etapa.

- Chaves públicas SSH para o AWS CodeCommit: selecione Fazer upload da chave pública SSH para carregar uma chave pública SSH para que o usuário possa se comunicar com os repositórios do CodeCommit via SSH.
- Credenciais do Git para o AWS CodeCommit: selecione Gerar credenciais para gerar um conjunto exclusivo de credenciais de usuário para uso com os repositórios Git. Selecione Baixar credenciais para salvar o nome de usuário e a senha em um arquivo .csv. Esse é o único momento em que essa informação está disponível. Se você esquecer ou perder a senha, precisará redefini-la.
- Credenciais para Amazon Keyspaces (para Apache Cassandra): selecione Gerar credenciais para gerar credenciais de usuário específicas do serviço para usar com o Amazon Keyspaces. Selecione Baixar credenciais para salvar o nome de usuário e a senha em um arquivo .csv. Esse é o único momento em que essa informação está disponível. Se você esquecer ou perder a senha, precisará redefini-la.

  > **Important**
  >
  > Credenciais específicas do serviço são credenciais de longo prazo associadas a um usuário do IAM específico e só podem ser usadas para o serviço para o qual foram criadas. Para conceder permissões aos perfis do IAM ou às identidades federadas para acessar todos os seus recursos do AWS usando credenciais temporárias, use a autenticação da AWS com o plug-in de autenticação SigV4 para o Amazon Keyspaces. Para obter mais informações, consulte Usar credenciais temporárias para se conectar ao Amazon Keyspaces (para Apache Cassandra) usando um perfil do IAM e o plug-in SigV4, no Guia do desenvolvedor do Amazon Keyspaces (para Apache Cassandra).

- Certificados de assinatura X.509: selecione Criar certificado X.509 se precisar fazer solicitações seguras do protocolo SOAP e estiver em uma região que não tem suporte pelo Gerenciador de certificados da AWS. O ACM é a ferramenta preferencial para provisionar, gerenciar e implantar seus certificados de servidor. Para obter mais informações sobre o ACM, consulte o Guia do usuário do Gerenciador de certificados da AWS.

Você criou um usuário com acesso programático e o configurou com a função de trabalho PowerUserAccess. A política de permissões desse usuário concede acesso total a todos os serviços, exceto para o IAM e o AWS Organizations.

Você pode usar esse mesmo processo para dar às workloads adicionais acesso programático aos seus recursos da Conta da AWS, caso as workloads não consigam assumir os perfis do IAM. Esse procedimento usou a política gerenciada PowerUserAccess para atribuir permissões. Para seguir a melhor prática de privilégio mínimo, considere usar uma política mais restritiva ou criar uma política personalizada que restrinja o acesso somente aos recursos exigidos pelo programa. Para saber mais sobre o uso de políticas que restringem as permissões do usuário a recursos específicos da AWS, consulte Gerenciamento de acesso para recursos da AWS e Exemplos de políticas baseadas em identidade do IAM. Para adicionar outros usuários ao grupo de usuários após a sua criação, consulte Editar usuários em grupos do IAM.

AWS CLI

1. Crie um usuário chamado **`Automation`**.

   - aws iam create-user

   ```
   aws iam create-user \
       --user-name Automation
   ```

2. Crie um grupo de usuários do IAM chamado **`AutomationGroup`**, anexe a política `PowerUserAccess` gerenciada pela AWS ao grupo e adicione o usuário **`Automation`** ao grupo.

   > **Note**
   >
   > Uma política gerenciada pela AWS é uma política independente que é criada e administrada pela AWS. Cada política tem seu próprio nome do recurso da Amazon

(ARN) que inclui o nome da política. Por exemplo, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` é uma política gerenciada da AWS. Para obter mais informações sobre ARNs do , consulte ARNs do IAM. Para obter uma lista das políticas gerenciadas pela AWS dos Serviços da AWS, consulte Políticas gerenciadas pela AWS.

- aws iam create-group

```
aws iam create-group \
    --group-name AutomationGroup
```

- aws iam attach-group-policy

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/PowerUserAccess \
    --group-name AutomationGroup
```

- aws iam add-user-to-group

```
aws iam add-user-to-group \
    --user-name Automation \
    --group-name AutomationGroup
```

- Execute o comando aws iam get-group para listar o **AutomationGroup** e seus membros.

```
aws iam get-group \
    --group-name AutomationGroup
```

3. Crie as credenciais de segurança necessárias para a workload.

- Crie chaves de acesso para testes: aws iam create-access-key

```
aws iam create-access-key \
```

```
                                --user-name Automation
```

A saída desse comando exibe a chave de acesso secreta e o ID da chave de acesso. Registre e armazene essas informações em um local seguro. Se essas credenciais forem perdidas, não será possível recuperá-las e você deverá criar uma nova chave de acesso.

> **Important**
>
> Essas chaves de acesso do usuário do IAM são credenciais de longo prazo que apresentam um risco de segurança para sua conta. Depois de concluir o teste, recomendamos excluir essas chaves de acesso. Se você tiver cenários em que esteja considerando chaves de acesso, investigue se você pode habilitar o MFA para seu usuário do IAM da workload e usar o aws sts get-session-token para obter credenciais temporárias para a sessão em vez de usar as chaves de acesso do IAM.

- Faça upload de chaves públicas SSH para o AWS CodeCommit: aws iam upload-ssh-public-key

  O exemplo a seguir pressupõe que você tenha suas chaves públicas SSH armazenadas no arquivo `sshkey.pub`.

  ```
                                aws upload-ssh-public-key \
                                    --user-name Automation \
                                    --ssh-public-key-body file://sshkey.pub
  ```

- Faça upload de um certificado de assinatura X.509: aws iam upload-signing-certificate

  Faça upload de um certificado X.509 se precisar fazer solicitações seguras do protocolo SOAP e estiver em uma região que não seja compatível com o Gerenciador de certificados da AWS. O ACM é a ferramenta preferencial para provisionar, gerenciar e implantar seus certificados de servidor. Para obter mais informações sobre o ACM, consulte o Guia do usuário do Gerenciador de certificados da AWS.

  O exemplo a seguir pressupõe que você tenha seu certificado de assinatura X.509 armazenado no arquivo `certificate.pem`.

```
aws iam upload-signing-certificate \
--user-name Automation \
--certificate-body file://certificate.pem
```

Você pode usar esse mesmo processo para dar às workloads adicionais acesso programático aos seus recursos da Conta da AWS, caso as workloads não consigam assumir os perfis do IAM. Esse procedimento usou a política gerenciada PowerUserAccess para atribuir permissões. Para seguir a melhor prática de privilégio mínimo, considere usar uma política mais restritiva ou criar uma política personalizada que restrinja o acesso somente aos recursos exigidos pelo programa. Para saber mais sobre o uso de políticas que restringem as permissões do usuário a recursos específicos da AWS, consulte Gerenciamento de acesso para recursos da AWS e Exemplos de políticas baseadas em identidade do IAM. Para adicionar outros usuários ao grupo de usuários após a sua criação, consulte Editar usuários em grupos do IAM.

Usar autenticação multifator com suas identidades

Usar autenticação multifator (MFA) com suas identidades é outra prática recomendada do IAM. A MFA é uma camada de segurança adicional que exige que os usuários forneçam fatores de autenticação adicionais após fornecerem seu nome de usuário e senha para verificar a identidade. Ela aumenta significativamente a segurança, tornando muito mais difícil para os invasores obterem acesso não autorizado, mesmo que a senha do usuário esteja comprometida. A MFA é amplamente adotada como uma prática recomendada para garantir o acesso a contas online, serviços em nuvem e outros recursos confidenciais. A AWS oferece suporte a MFA para o usuário-raiz, usuários do IAM, usuários no Centro de Identidade do IAM, ID do builder e usuários federados. Para segurança adicional, você pode criar políticas que exijam que a MFA seja configurada antes de permitir que um usuário acesse recursos ou realize ações específicas e anexe essas políticas aos perfis do IAM. O Centro de Identidade do IAM é pré-configurado com a MFA ativada por padrão para que todos os usuários no Centro de Identidade do IAM precisem iniciar sessão com a MFA além do nome de usuário e a senha.

> **Note**
>
> Todos os tipos de Conta da AWS (contas autônomas, de gerenciamento e de membros) exigem que a MFA seja configurada para o usuário-raiz. Os usuários precisarão registrar

a MFA em até 35 dias após a primeira tentativa de login, para acessar o Console de gerenciamento da AWS, se a MFA ainda não estiver habilitada.

Para obter mais informações, consulte Configurar MFA no Centro de Identidade do IAM e AWSCódigo da autenticação multifator no IAM da.

Preparar para permissões de privilégio mínimo

Usar permissões com privilégio mínimo é uma indicação de prática recomendada do IAM. O conceito de permissões de privilégio mínimo é conceder aos usuários somente as permissões necessárias para realizar uma tarefa. Ao configurar, considere como você oferecerá suporte às permissões de privilégio mínimo. O usuário-raiz, o usuário administrador e o usuário de acesso de emergência do IAM têm permissões avançadas que não são necessárias para tarefas diárias. Enquanto você está aprendendo sobre a AWS e testando diferentes serviços, recomendamos criar pelo menos um usuário adicional no Centro de Identidade do IAM com permissões menores, que possa ser usado em diferentes cenários. É possível usar as políticas do IAM para definir as ações que podem ser executadas em recursos específicos sob condições específicas e se conectar a esses recursos com sua conta de privilégio menor.

Se você estiver usando o Centro de Identidade do IAM, considere usar conjuntos de permissões dele para começar. Para saber mais, consulte Criar um conjunto de permissões no Guia do usuário do Centro de Identidade do IAM.

Caso não esteja usando o Centro de Identidade do IAM, use perfis do IAM para definir as permissões para diferentes entidades do IAM. Para saber mais, consulte Criar um perfil do IAM.

Tanto os perfis do IAM como os conjuntos de permissões do Centro de Identidade do IAM podem usar políticas gerenciadas pela AWS com base em funções de trabalho. Para obter detalhes sobre as permissões concedidas por essas políticas, consulte AWSPolíticas gerenciadas pela para funções de trabalho.

Important

Lembre-se de que as políticas gerenciadas pela AWS podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque estão disponíveis para uso por todos os clientes da AWS. Depois de configurar, recomendamos usar o IAM Access Analyzer para gerar políticas de privilégio mínimo com base na atividade de acesso que está

registrada no AWS CloudTrail. Para obter mais informações sobre a geração de políticas, consulte Geração de políticas do IAM Access Analyzer.

Ao começar, recomendamos usar políticas gerenciadas pela AWS para conceder permissões. Depois de um período de amostra de atividade predefinido (por exemplo, 90 dias), você poderá revisar os serviços que as pessoas e workloads acessaram. Em seguida, você poderá criar uma nova política gerenciada pelo cliente com permissões reduzidas para substituir a política gerenciada pela AWS. A nova política deve incluir somente os serviços que foram acessados durante o período da amostra. Atualize suas permissões para remover a política gerenciada pela AWS e anexe a nova política gerenciada pelo cliente que você criou.

Para obter instruções sobre como refinar as permissões, consulte Revisar informações acessadas por último da sua conta da AWS. Depois de identificar os serviços não utilizados, consulte Gerar uma política com base na atividade de acesso.

Revisar informações acessadas por último da sua conta da AWS

Você pode visualizar as informações sobre o último acesso ao serviço para o IAM usando o console do IAM, a AWS CLI ou a API da AWS. Para obter informações importantes sobre os dados, permissões necessárias, solução de problemas e regiões com suporte, consulte Refinar permissões na AWS usando informações do último acesso.

É possível visualizar as informações para os tipos de recurso a seguir no IAM. Em cada caso, as informações abrangem serviços permitidos para o período de geração de relatórios indicado:

- Usuário do IAM: visualize a última vez em que o usuário tentou acessar cada serviço permitido.
- Grupo do IAM: visualize as informações sobre a última vez em que um membro do grupo do IAM tentou acessar cada serviço permitido. Esse relatório também inclui o número total de membros que tentou acessar.
- Perfil do IAM: visualize a última vez em que alguém usou o perfil em uma tentativa de acessar cada serviço permitido.
- Política: visualize informações sobre a última vez em que um usuário ou uma função tentou acessar cada serviço permitido. Esse relatório também inclui o número total de entidades que tentou acessar.

> **Note**
>
> Para visualizar as informações de acesso de um recurso no IAM, compreenda o período de geração de relatórios, as entidades relatadas e os tipos de política avaliados para as informações. Para obter mais detalhes, consulte the section called “Coisas para saber sobre as informações acessadas por último”.

Para obter mais informações sobre as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Para revisar as informações acessadas por último em uma Conta da AWS

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha User groups (Grupos de usuários), Users (USuários), Roles (Funções) ou Policies (Políticas).
4. Escolha qualquer nome de usuário, grupo de usuários, perfil ou política para abrir a página Resumo correspondente e escolha a guia Acessados recentemente. Exiba as seguintes informações com base no recurso escolhido:

   - User group (Grupo de usuários): exibe a lista de serviços que os membros do grupo de usuários podem acessar. Você também pode visualizar quando um membro acessou o serviço pela última vez, quais políticas de grupo de usuário ele usou e qual membro do grupo de usuários fez a solicitação. Escolha o nome da política para saber se ela é uma política gerenciada ou uma política de grupo de usuários em linha. Escolha o nome do membro do grupo de usuários para ver todos os membros do grupo de usuários e quando eles acessaram o serviço mais recentemente.
   - User (Usuário): exibe a lista de serviços que o usuário pode acessar. Também é possível exibir quando ele acessou o serviço pela última vez e quais políticas estão associadas ao usuário no momento. Escolha o nome da política para saber se é uma política gerenciada, uma política de usuário em linha ou uma política em linha para o grupo.

- Role (Função): exibe a lista de serviços que a função pode acessar, quando a função acessou o serviço mais recentemente e quais políticas foram usadas. Escolha o nome da política para saber se ela é uma política gerenciada ou uma política de função em linha.
- Policy (Política): exibe a lista de serviços com ações permitidas na política. Você também pode exibir quando a política foi usada pela última vez para acessar o serviço e qual entidade (usuário ou função) usou a política. A data de Último acesso também inclui quando o acesso é concedido a essa política por meio de outra política. Escolha o nome da entidade para saber quais entidades têm essa política anexada e quando elas acessaram o serviço mais recentemente.

5. Na coluna Serviço da tabela, escolha o nome de um dos serviços que inclui informações acessadas pela última vez pela ação para visualizar uma lista de ações de gerenciamento que as entidades do IAM tentaram acessar. É possível visualizar a Região da AWS e um carimbo de data/hora que mostre quando alguém tentou executar a ação pela última vez.
6. A coluna Último acesso é exibida para serviços e ações de gerenciamento dos serviços que incluem informações acessadas pela última vez pela ação. Revise os seguintes resultados possíveis que são retornados nessa coluna. Esses resultados variam dependendo se um serviço ou ação é permitido, foi acessado e se ele é rastreado pela AWS quanto a informações acessadas por último.

   <number of> dias atrás

   O número de dias desde que o serviço ou a ação foi usado no período de rastreamento. O período de rastreamento dos serviços é para os últimos 400 dias. O período de rastreamento das ações do Amazon S3 começou em 12 de abril de 2020. O período de rastreamento das ações do Amazon EC2, do IAM e do Lambda começou em 7 de abril de 2021. O período de rastreamento para todos os outros serviços começou em 23 de maio de 2023. Para saber mais sobre as datas de início de rastreamento para cada Região da AWS, consulte Onde a AWS rastreia informações acessadas por último.

   Não acessado no período de rastreamento

   O serviço ou ação rastreado não foi usado por uma entidade no período de rastreamento.

   É possível que você tenha permissões para uma ação que não aparece na lista. Isso poderá acontecer se as informações de rastreamento da ação não tiverem sido incluídas pela AWS. Você não deve tomar decisões de permissões com base apenas na ausência de informações de rastreamento. Em vez disso, recomendamos que você use essas informações para

informar e apoiar sua estratégia geral de concessão de privilégios mínimos. Verifique suas políticas para confirmar se o nível de acesso é apropriado.

AWS CLI

Você pode usar a AWS CLI para recuperar informações sobre a última vez que um recurso do IAM em sua Conta da AWS foi usado para tentar acessar serviços da AWS e ações do Amazon S3, Amazon EC2, IAM e Lambda. Um recurso do IAM pode ser um usuário, grupo de usuários, função ou política.

- Gere um relatório para recursos do IAM em uma Conta da AWS. A solicitação deve incluir o ARN do recurso do IAM (usuário, grupo de usuários, função ou política) para o qual você deseja um relatório. Você pode especificar o nível de granularidade que deseja gerar no relatório para exibir detalhes de acesso para serviços ou serviços e ações. A solicitação retorna um `job-id` que você pode acabar usando nas operações `get-service-last-accessed-details` e `get-service-last-accessed-details-with-entities` para monitorar o `job-status` até o trabalho estar concluído.

  - aws iam generate-service-last-accessed-details

  a. Recupere detalhes sobre o relatório usando o parâmetro `job-id` da etapa anterior.

     - aws iam get-service-last-accessed-details

     Essa operação retorna as seguintes informações, com base no tipo de recurso e nível de granularidade que você solicitou na operação `generate-service-last-accessed-details`:

     - Usuário: retorna uma lista de serviços que o usuário especificado pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa do usuário e o ARN do usuário.
     - Grupo de usuários:retorna uma lista de serviços que os membros do grupo de usuários especificado podem acessar usando políticas anexadas ao grupo de usuários. Para cada serviço, a operação retorna a data e a hora da última tentativa feita por qualquer membro do grupo de usuários. Ela também retorna o ARN do usuário e o número total de membros do grupode usuários que tentou acessar o

serviço. Use a operação GetServiceLastAccessedDetailsWithEntities para recuperar uma lista de todos os membros.

- Função: retorna uma lista de serviços que a função especificada pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa da função e o ARN da função.
- Política: retorna uma lista de serviços para os quais a política especificada permite o acesso. Para cada serviço, a operação retorna a data e a hora em que uma entidade (usuário ou função) tentou acessar mais recentemente o serviço usando a política. Ele também retorna o ARN dessa entidade e o número total de entidades que tentou acessar.

b. Saiba mais sobre as entidades que usaram permissões de grupo de usuários ou política em uma tentativa de acessar um serviço específico. Essa operação retorna uma lista de entidades com ARN, ID, nome, caminho, tipo (usuário ou função) de cada entidade e quando eles tentaram acessar o serviço mais recentemente. Você também pode usar essa operação para usuários e funções, mas ela só retorna informações sobre essa entidade.

   - aws iam get-service-last-accessed-details-with-entities

c. Saiba mais sobre as políticas baseadas em identidade que uma identidade (usuário, grupo de usuários ou função) usou em uma tentativa de acessar um serviço específico. Quando você especifica uma identidade e um serviço, essa operação retorna uma lista de políticas de permissões que a identidade pode usar para acessar o serviço especificado. Essa operação indica o estado atual de políticas e não depende do relatório gerado. Isso também não retorna outros tipos de política, como políticas baseadas em recurso, listas de controle de acesso, políticas do AWS Organizations, limites de permissões do IAM ou políticas de sessão. Para ter mais informações, consulte Tipos de políticas ou Avaliação de políticas para solicitações em uma única conta.

   - aws iam list-policies-granting-service-access

API

Você pode usar a API da AWS para recuperar informações sobre a última vez que um recurso do IAM foi usado para tentar acessar produtos da AWS e ações do Amazon S3, do Amazon EC2, do IAM e do Lambda. Um recurso do IAM pode ser um usuário, grupo de usuários, função

ou política. Você pode especificar o nível de granularidade a ser gerado no relatório para exibir detalhes de serviços ou serviços e ações.

1. Gere um relatório. A solicitação deve incluir o ARN do recurso do IAM (usuário, grupo de usuários, função ou política) para o qual você deseja um relatório. Ele retorna um `JobId` que você pode acabar usando nas operações `GetServiceLastAccessedDetails` e `GetServiceLastAccessedDetailsWithEntities` para monitorar o `JobStatus` até o trabalho estar concluído.

   - GenerateServiceLastAccessedDetails

2. Recupere detalhes sobre o relatório usando o parâmetro `JobId` da etapa anterior.

   - GetServiceLastAccessedDetails

   Essa operação retorna as seguintes informações, com base no tipo de recurso e nível de granularidade que você solicitou na operação `GenerateServiceLastAccessedDetails`:

   - Usuário: retorna uma lista de serviços que o usuário especificado pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa do usuário e o ARN do usuário.
   - Grupo de usuários:retorna uma lista de serviços que os membros do grupo de usuários especificado podem acessar usando políticas anexadas ao grupo de usuários. Para cada serviço, a operação retorna a data e a hora da última tentativa feita por qualquer membro do grupo de usuários. Ela também retorna o ARN do usuário e o número total de membros do grupode usuários que tentou acessar o serviço. Use a operação GetServiceLastAccessedDetailsWithEntities para recuperar uma lista de todos os membros.
   - Função: retorna uma lista de serviços que a função especificada pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa da função e o ARN da função.
   - Política: retorna uma lista de serviços para os quais a política especificada permite o acesso. Para cada serviço, a operação retorna a data e a hora em que uma entidade (usuário ou função) tentou acessar mais recentemente o serviço usando a política. Ele também retorna o ARN dessa entidade e o número total de entidades que tentou acessar.

3. Saiba mais sobre as entidades que usaram permissões de grupo de usuários ou política em uma tentativa de acessar um serviço específico. Essa operação retorna uma lista de

entidades com ARN, ID, nome, caminho, tipo (usuário ou função) de cada entidade e quando eles tentaram acessar o serviço mais recentemente. Você também pode usar essa operação para usuários e funções, mas ela só retorna informações sobre essa entidade.

- GetServiceLastAccessedDetailsWithEntities

4. Saiba mais sobre as políticas baseadas em identidade que uma identidade (usuário, grupo de usuários ou função) usou em uma tentativa de acessar um serviço específico. Quando você especifica uma identidade e um serviço, essa operação retorna uma lista de políticas de permissões que a identidade pode usar para acessar o serviço especificado. Essa operação indica o estado atual de políticas e não depende do relatório gerado. Isso também não retorna outros tipos de política, como políticas baseadas em recurso, listas de controle de acesso, políticas do AWS Organizations, limites de permissões do IAM ou políticas de sessão. Para ter mais informações, consulte Tipos de políticas ou Avaliação de políticas para solicitações em uma única conta.

   - ListPoliciesGrantingServiceAccess

Gerar uma política com base na atividade de acesso

Você pode usar a atividade de acesso registrada no AWS CloudTrail para um usuário do IAM ou perfil do IAM para que o IAM Access Analyzer gere uma política gerenciada pelo cliente para permitir acesso apenas aos serviços de que usuários e perfis específicos precisam.

Quando IAM Access Analyzer gera uma política do IAM, as informações são retornadas para ajudar você a personalizar ainda mais a política. Duas categorias de informações podem ser retornadas quando uma política é gerada:

- Policy with action-level information (Política com informações no nível de ação): para alguns produtos da AWS, como o Amazon EC2, o IAM o Access Analyzer podem identificar as ações encontradas nos eventos do CloudTrail e listar as ações usadas na política gerada. Para obter uma lista dos serviços compatíveis, consulte Serviços de geração de política do IAM Access Analyzer. Para alguns serviços, o IAM Access Analyzer solicita que você adicione ações para os serviços à política gerada.
- A política com informações em nível de serviço – IAM Access Analyzer usa as últimas informações acessadas para criar um modelo de política com todos os serviços usados recentemente. Ao usar o Console de gerenciamento da AWS, solicitamos que você revise os serviços e adicione ações para concluir a política.

Para gerar uma política com base na atividade de acesso

No procedimento seguinte, reduziremos as permissões concedidas a um perfil para corresponder ao uso de um usuário. Ao escolher um usuário, escolha um usuário cujo uso exemplifique o perfil. Muitos clientes configuram contas de usuário de teste com permissões PowerUser e, em seguida, fazem com que realizem um conjunto específico de tarefas por um curto período de tempo para determinar qual acesso é necessário para realizar essas tarefas.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Usuários e, em seguida, escolha o nome de usuário para acessar a página de detalhes do usuário.
4. Na guia Permissões, em Gerar política com base em eventos do CloudTrail, escolha Gerar política.
5. Na página Gerar política, configure os seguintes itens:
   - Em Selecionar período, escolha Últimos sete dias.
   - Para Trilha do CloudTrail a ser analisada, selecione a região e a trilha em que a atividade desse usuário é registrada.
   - Escolha Criar e usar um novo perfil de serviço.
6. Escolha Gerar política e aguarde até que o perfil seja criado. Não atualize nem saia da página do console até que a mensagem de notificação Geração de política em andamento apareça.
7. Depois que a política for gerada, você deverá analisá-la e personalizá-la conforme necessário com os IDs da conta e os ARNs dos recursos. Além disso, a política gerada automaticamente pode não incluir as informações de nível de ação necessárias para concluir a política. Para obter mais informações, consulte Geração de políticas do analisador de acesso do IAM.

   Por exemplo, você pode editar a primeira instrução que inclui o efeito `Allow` e o elemento `NotAction` para permitir apenas ações do Amazon EC2 e Amazon S3. Para isso, substitua-

a pela instrução com o ID `FullAccessToSomeServices`. A nova política pode ser semelhante à seguinte política de exemplo.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Sid": "FullAccessToSomeServices",
          "Effect": "Allow",
          "Action": [
              "ec2:*",
              "s3:*"
          ],
          "Resource": "*"
      },
      {
          "Effect": "Allow",
          "Action": [
              "iam:CreateServiceLinkedRole",
              "iam:DeleteServiceLinkedRole",
              "iam:ListRoles",
              "organizations:DescribeOrganization"
          ],
          "Resource": "*"
      }
  ]
}
```

8. Para apoiar a prática recomendada de concessão de privilégios mínimos, revise e corrija quaisquer erros, avisos ou sugestões retornados durante a validação da política.
9. Para reduzir ainda mais as permissões das políticas para ações e recursos específicos, exiba os eventos no Event history (Histórico de eventos) do CloudTrail. Lá você pode exibir informações detalhadas sobre as ações e os recursos específicos acessados pelo usuário. Para obter mais informações, consulte Visualizar eventos do CloudTrail no console do CloudTrail no Guia do usuário do AWS CloudTrail.
10. Depois de revisar e validar sua política, salve-a com um nome descritivo.
11. Navegue até a página Perfis e escolha o perfil que as pessoas assumirão quando realizarem as tarefas permitidas pela sua nova política.

12. Selecione a guia Permissões e depois Adicionar permissões e selecione Anexar políticas.
13. Na página Anexar políticas de permissão, na lista Outras políticas de permissões, selecione a política que você criou anteriormente e escolha Anexar políticas.
14. Você retorna à página de detalhes de Perfil. Há duas políticas associadas ao perfil, sua política previamente gerenciada pela AWS, como PowerUserAccess, e sua nova política. Marque a caixa de seleção da política gerenciada pela AWS e escolha Remover. Quando solicitado a confirmar a remoção, escolha Remover.

Os usuários do IAM, entidades principais de usuários federados SAML e OIDC e workloads que assumem esse perfil agora têm acesso reduzido de acordo com a nova política que você criou.

AWS CLI

Você pode usar os seguintes comandos para gerar uma política usando o AWS CLI.

Para gerar uma política

- aws accessanalyzer start-policy-generation

Para exibir uma política gerada

- aws accessanalyzer get-generated-policy

Para cancelar uma solicitação de geração de política

- aws accessanalyzer cancel-policy-generation

Para exibir uma lista de solicitações de geração de políticas

- aws accessanalyzer list-policy-generations

API

Você pode usar as seguintes operações para gerar uma política usando a API AWS.

Para gerar uma política

- StartPolicyGeneration

Para exibir uma política gerada

- GetGeneratedPolicy

Para cancelar uma solicitação de geração de política

- CancelPolicyGeneration

Para exibir uma lista de solicitações de geração de políticas

- ListPolicyGenerations

Usar pesquisa para encontrar recursos do IAM

Ao analisar suas descobertas de acesso, você pode usar a página de pesquisa do console do IAM como uma opção mais rápida para localizar recursos do IAM. Você pode pesquisar recursos usando nomes de recursos parciais ou ARNs.

Console

O recurso de pesquisa do console do IAM pode localizar qualquer um dos seguintes itens:

- Nomes de entidades do IAM que correspondem às suas palavras-chave de pesquisa (para usuários, grupos, funções, provedores de identidade e políticas)
- Tarefas que atendem às suas palavras-chave de pesquisa

O recurso de pesquisa do console do IAM não retorna informações sobre o IAM Access Analyzer.

Cada linha no resultado de pesquisa é um link ativo. Por exemplo, você pode escolher o nome de usuário no resultado de pesquisa, o que o leva à página de detalhes desse usuário. Ou você pode escolher um link de ação, por exemplo, Criar usuário, para ir para a página Criar usuário.

> **Note**
>
> A pesquisa de chave de acesso requer que você digite o ID de chave de acesso na caixa de pesquisa. O resultado da pesquisa mostra o usuário associado a essa chave. A partir

daí, você pode acessar diretamente a página desse usuário, onde pode gerenciar a chave de acesso.

Use a página Search (Pesquisar) no console do IAM para encontrar itens relacionados a essa conta.

Para procurar itens no console do IAM

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione Pesquisar.
4. Na caixa Search (Pesquisar), digite a palavra-chave de pesquisa.
5. Escolha um link na lista de resultados da pesquisa para acessar a parte correspondente do console.

Os ícones a seguir identificam os tipos de itens que são encontradas por uma pesquisa:

Ícone	Descrição
	Usuários do IAM
	Grupos do IAM
	Perfis do IAM
	Políticas do IAM
	Tarefas como "criar usuário" ou "anexar política"

Ícone	Descrição
	Resultados da palavra-chave **delete**

Exemplos de frases de pesquisa

Você pode usar as seguintes frases na pesquisa do IAM. Substitua os termos em itálico pelos nomes dos verdadeiros usuários, grupos, perfis, chaves de acesso, políticas ou provedores de identidade do IAM que você deseja localizar.

- ***user_name*** ou ***group_name*** ou ***role_name*** ou ***policy_name*** ou ***identity_provider_name***
- ***access_key***
- **add user *user_name* to groups** ou **add users to group *group_name***
- **remove user *user_name* from groups**
- **delete *user_name*** ou **delete *group_name*** ou **delete *role_name*** ou **delete *policy_name*** ou **delete *identity_provider_name***
- **manage access keys *user_name***
- **manage signing certificates *user_name***
- **users**
- **manage MFA for *user_name***
- **manage password for *user_name***
- **create role**
- **password policy**
- **edit trust policy for role *role_name***
- **show policy document for role *role_name***
- **attach policy to *role_name***
- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***
- **attach entities to *policy_name***

- **detach entities from *policy_name***

Melhores práticas de segurança e casos de uso no AWS Identity and Access Management

O AWS Identity and Access Management (IAM) fornece vários recursos de segurança que devem ser considerados no desenvolvimento e na implementação das suas próprias políticas de segurança. As melhores práticas a seguir são diretrizes gerais e não representam uma solução completa de segurança.

Essas práticas recomendadas podem não ser apropriadas ou suficientes para seu ambiente específico. Portanto, trate-as como considerações úteis, e não como requisitos.

Para aproveitar ao máximo o IAM, reserve tempo para conhecer as práticas recomendadas. Uma forma de fazer isso é ver como o IAM é usado em cenários reais para trabalhar com outros Serviços da AWS.

Tópicos

- Práticas recomendadas de segurança no IAM
- As práticas recomendadas do usuário-raiz para o Conta da AWS
- Casos de uso de negócios do IAM

Práticas recomendadas de segurança no IAM

Follow us on Twitter

Para ajudar a proteger seus recursos da AWS, siga estas práticas recomendadas para o AWS Identity and Access Management (IAM).

Depois de implementar essas melhores práticas, consulte Segurança no IAM e no AWS STS para obter tópicos adicionais de segurança, incluindo validação de conformidade, resiliência e segurança da infraestrutura.

Tópicos

- Exija que os usuários humanos usem a federação com um provedor de identidade para acessar a AWS usando credenciais temporárias
- Exija que as workloads usem credenciais temporárias com perfis do IAM para acessar a AWS
- Exija autenticação multifator (MFA)

- Atualize as chaves de acesso quando necessário para casos de uso que exijam credenciais de longo prazo
- Siga as melhores práticas para proteger as credenciais do usuário-raiz
- Aplique permissões de privilégio mínimo
- Comece com as políticas gerenciadas pela AWS e avance para as permissões de privilégio mínimo.
- Use o IAM Access Analyzer para gerar políticas de privilégios mínimos com base na atividade de acesso
- Revise e remova regularmente usuários, funções, permissões, políticas e credenciais não utilizados
- Use condições nas políticas do IAM para restringir ainda mais o acesso
- Verifique o acesso entre contas e público aos recursos com o IAM Access Analyzer
- Use o analisador de acesso do IAM para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais
- Estabeleça barreiras de proteção para permissões em várias contas
- Use limites de permissões para delegar o gerenciamento de permissões em uma conta

Exija que os usuários humanos usem a federação com um provedor de identidade para acessar a AWS usando credenciais temporárias

Os usuários humanos, também conhecidos como identidades humanas, são as pessoas, os administradores, os desenvolvedores, os operadores e os consumidores de suas aplicações. Eles devem ter uma identidade para acessar seus ambientes e aplicações da AWS. Os usuários humanos que são membros de sua organização também são conhecidos como identidades de força de trabalho. Os usuários humanos também podem ser usuários externos com quem você colabora e que interagem com seus recursos da AWS. Eles podem fazer isso por meio de um navegador da web, aplicativo do cliente, aplicativo para dispositivo móvel ou ferramentas interativas de linha de comando.

Exija que seus usuários humanos usem credenciais temporárias ao acessar a AWS. É possível usar um provedor de identidade para que seus usuários humanos recebam acesso federado a Contas da AWS assumindo perfis que fornecem credenciais temporárias. Para gerenciamento de acesso centralizado, recomendamos que você usar o Centro de Identidade do AWS IAM (IAM Identity Center) para gerenciar o acesso às suas contas e as permissões nessas contas. É possível

gerenciar suas identidades de usuário com o IAM Identity Center ou gerenciar permissões de acesso para identidades de usuário no IAM Identity Center de um provedor de identidade externo. Para obter mais informações, consulte O que é o Centro de Identidade do AWS IAM no Guia do usuário do Centro de Identidade do AWS IAM.

Você pode usar o gerenciador de acesso à conta — um atributo do IAM que permite atribuir perfis do IAM existentes nas contas da sua organização aos usuários e grupos do IAM Identity Center. O gerenciador de acesso à conta fornece acesso ao conjunto completo de atributos de perfis do IAM, incluindo políticas de confiança personalizadas, tags de perfil para ABAC e caminhos de perfil configuráveis. Você pode usá-lo junto com os conjuntos de permissões ou isoladamente.

Para ter mais informações sobre funções, consulte Termos e conceitos das funções.

Exija que as workloads usem credenciais temporárias com perfis do IAM para acessar a AWS

Uma workload é uma coleção de códigos e recursos que fornece valor comercial, como uma aplicação ou um processo de backend. Sua workload pode ter aplicações, ferramentas operacionais e componentes que exigem credenciais para fazer solicitações aos Serviços da AWS, como solicitações de leitura de dados do Amazon S3.

Quando você está criando em um serviço de computação da AWS, como Amazon EC2 ou Lambda, a AWS entrega as credenciais temporárias de um perfil do IAM a esse recurso computacional. As aplicações criadas usando um AWS SDK descobrirão e usarão essas credenciais temporárias para acessar recursos da AWS, e não há necessidade de distribuir credenciais de longa duração de um usuário do IAM para suas workloads em execução na AWS.

As workloads executadas fora da AWS, como seus servidores on-premises, servidores de outros provedores de nuvem ou plataformas gerenciadas de integração contínua e entrega contínua (CI/CD), ainda podem usar credenciais temporárias. No entanto, você precisará fornecer essas credenciais temporárias à sua workload. Algumas maneiras de fornecer credenciais temporárias às suas workloads são:

- Você pode usar o IAM Roles Anywhere para solicitar credenciais da AWS temporárias para sua workload usando um certificado X.509 da sua infraestrutura de chave pública (PKI).
- É possível chamar a API `AssumeRoleWithSAML` do AWS STS para solicitar credenciais temporárias da AWS para sua workload usando uma declaração SAML de um provedor de identidades (IdP) externo configurado dentro da sua Conta da AWS.

- É possível chamar a API `AssumeRoleWithWebIdentity` do AWS STS para solicitar credenciais temporárias da AWS para sua workload usando um token Web JSON (JWT) de um IdP configurado em sua Conta da AWS.
- É possível solicitar credenciais temporárias da AWS no seu dispositivo de IoT usando autenticação Mutual Transport Layer Security (MTLS) com AWS IoT Core.

Alguns Serviços da AWS também oferecem suporte a integrações para fornecer credenciais temporárias para suas workloads executadas fora da AWS:

- O Amazon Elastic Container Service (Amazon ECS) Anywhere permite executar tarefas do Amazon ECS em seus próprios recursos computacionais e fornece credenciais da AWS temporárias para suas tarefas do Amazon ECS executadas nesses recursos computacionais.
- O Amazon Elastic Kubernetes Service Hybrid Nodes permite adicionar seus recursos computacionais executados fora da AWS como nós a um cluster do Amazon EKS. O Amazon EKS pode fornecer credenciais temporárias aos pods do Amazon EKS executados em seus recursos computacionais.
- O AWS Systems ManagerHybrid Activations permite gerenciar seus recursos computacionais executados fora da AWS usando o SSM e fornece credenciais da AWS temporárias ao agente SSM executado em seus recursos computacionais.

Exija autenticação multifator (MFA)

Recomendamos o uso de perfis do IAM para usuários humanos e workloads que acessam seus recursos da AWS a fim de que usem credenciais temporárias. No entanto, para cenários em que você precisa de um usuário do IAM ou usuário-raiz em sua conta, exija MFA para aumentar a segurança. Com a MFA, os usuários têm um dispositivo que gera uma resposta a um desafio de autenticação. As credenciais de cada usuário e a resposta gerada pelo dispositivo são necessárias para concluir o processo de login. Para obter mais informações, consulte AWSCódigo da autenticação multifator no IAM da. Onde possível, recomendamos usar MFA resistente a phishing, como chaves de acesso e chaves de segurança. Para obter mais informações, consulte Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS.

Se você usa o IAM Identity Center para gerenciamento de acesso centralizado para usuários humanos, poderá usar os recursos de MFA do IAM Identity Center quando sua origem de identidade estiver configurada com o armazenamento de identidade do IAM Identity Center, do AWS Managed Microsoft AD ou do AD Connector. Para obter mais informações sobre a MFA no IAM Identity Center,

consulte Multi-factor authentication (Autenticação multifator) no Guia do usuário do Centro de Identidade do AWS IAM.

Atualize as chaves de acesso quando necessário para casos de uso que exijam credenciais de longo prazo

Sempre que possível, recomendamos depender de credenciais temporárias em vez de criar credenciais de longo prazo, como chaves de acesso. No entanto, para cenários em que você precise de usuários do IAM com acesso programático e credenciais de longo prazo, recomendamos atualizar as chaves de acesso sempre que necessário, por exemplo, quando um funcionário sair da empresa. Recomendamos usar as últimas informações de acesso do IAM para atualizar e remover as chaves de acesso com segurança. Para obter mais informações, consulte Atualizar chaves de acesso.

Há casos de uso específicos que exigem credenciais de longo prazo com usuários do IAM na AWS. Alguns dos casos de uso incluem o seguinte:

- Workloads que não podem usar perfis do IAM: você pode executar a workload de um local que precisa acessar a AWS. Em algumas situações, você não pode usar perfis do IAM para fornecer credenciais temporárias, como para plugins do WordPress. Nessas situações, use as chaves de acesso de longo prazo do usuário do IAM para que essa workload seja autenticada na AWS.
- Clientes de terceiros da AWS: se você estiver usando ferramentas que não oferecem suporte ao IAM Identity Center, como clientes da AWS de terceiros ou fornecedores que não estão hospedados na AWS, use as chaves de acesso de longo prazo do usuário do IAM.
- Acesso ao AWS CodeCommit: se você estiver usando o CodeCommit para armazenar seu código, poderá usar um usuário do IAM com chaves SSH ou credenciais específicas de serviço para que o CodeCommit seja autenticado em seus repositórios. Recomendamos fazer isso além de usar um usuário do IAM Identity Center para autenticação padrão. Os usuários do Centro de Identidade do IAM são as pessoas em sua força de trabalho que precisam acessar suas Contas da AWS ou suas aplicações na nuvem. Para dar aos usuários acesso aos seus repositórios do CodeCommit sem configurar os usuários do IAM, você pode configurar o utilitário git-remote-codecommit. Para obter mais informações sobre o IAM e o CodeCommit, consulte Credenciais do IAM para o CodeCommit: credenciais do Git, chaves SSH e chaves de acesso da AWS. Para obter mais informações sobre como configurar o utilitário git-remote-codecommit, consulte Conectar-se a repositórios do AWS CodeCommit credenciais alternadas, no Guia do usuário do AWS CodeCommit.
- Acesso ao Amazon Keyspaces (para Apache Cassandra): em uma situação em que não é possível usar usuários no IAM Identity Center, como para fins de teste de compatibilidade com

o Cassandra, você pode usar um usuário do IAM com credenciais específicas do serviço para a autenticação com o Amazon Keyspaces. Os usuários do Centro de Identidade do IAM são as pessoas em sua força de trabalho que precisam acessar suas Contas da AWS ou suas aplicações na nuvem. Você também pode se conectar ao Amazon Keyspaces usando credenciais temporárias. Para obter mais informações, consulte Using temporary credentials to connect to Amazon Keyspaces using an IAM role and the SigV4 plugin (Como usar credenciais temporárias para se conectar ao Amazon Keyspaces usando um perfil do IAM e o plug-in SigV4) no Guia do desenvolvedor do Amazon Keyspaces (para Apache Cassandra).

Siga as melhores práticas para proteger as credenciais do usuário-raiz

Ao criar uma Conta da AWS, você estabelece as credenciais do usuário-raiz para fazer login no Console de gerenciamento da AWS. Proteja suas credenciais de usuário-raiz da mesma forma como protegeria outras informações pessoais confidenciais. Para entender melhor como proteger e escalar seus processos de usuário-raiz, consulte As práticas recomendadas do usuário-raiz para o Conta da AWS.

Aplique permissões de privilégio mínimo

Ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. É possível começar com permissões amplas enquanto explora as permissões necessárias para sua workload ou para seu caso de uso. À medida que seu caso de uso se desenvolve, você pode trabalhar para reduzir as permissões que concede para caminhar em direção ao privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no AWS Identity and Access Management.

Comece com as políticas gerenciadas pela AWS e avance para as permissões de privilégio mínimo.

Para começar a conceder permissões a seus usuários e workloads, use as políticas gerenciadas pela AWS que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis na sua Conta da AWS. Lembre-se de que as políticas gerenciadas pela AWS podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque estão disponíveis para uso por todos os clientes da AWS. Como resultado, recomendamos que você reduza ainda mais as permissões definindo as políticas gerenciadas pelo cliente específicas para seus casos de uso. Para

obter mais informações, consulte AWSPolíticas gerenciadas pela. Para obter mais informações sobre políticas gerenciadas pela AWS que são projetadas para funções de trabalho específicas, consulte AWSPolíticas gerenciadas pela para funções de trabalho.

Use o IAM Access Analyzer para gerar políticas de privilégios mínimos com base na atividade de acesso

Para conceder apenas as permissões necessárias para executar uma tarefa, você pode gerar políticas com base em sua atividade de acesso registrada no AWS CloudTrail. O IAM Access Analyzer analisa os serviços e as ações que seus perfis do IAM usam e, em seguida, gera uma política aperfeiçoada que você pode utilizar. Depois de testar cada política gerada, você pode implantar a política em seu ambiente de produção. Isso garante que você conceda apenas as permissões necessárias para suas workloads. Para obter mais informações sobre a geração de políticas, consulte Geração de políticas do IAM Access Analyzer.

Revise e remova regularmente usuários, funções, permissões, políticas e credenciais não utilizados

É possível ter usuários, perfis, permissões, políticas ou credenciais do IAM de que não precisa mais em sua Conta da AWS. O IAM fornece as informações do seu último acesso para ajudar você a identificar os usuários, os perfis, as permissões, as políticas e as credenciais que não são mais necessários para que possa removê-los. Isso ajuda a reduzir o número de usuários, perfis, permissões, políticas e credenciais que você precisa monitorar. Você também pode usar essas informações para aperfeiçoar suas políticas do IAM para aderir melhor às permissões de privilégio mínimo. Para obter mais informações, consulte Refinar permissões na AWS usando informações do último acesso.

Use condições nas políticas do IAM para restringir ainda mais o acesso

É possível especificar as condições sob as quais uma declaração de política está em vigor. Dessa forma, você pode conceder acesso a ações e recursos, mas somente se a solicitação de acesso atender a condições específicas. Por exemplo, você pode escrever uma condição de política para especificar que todas as solicitações devem ser enviadas via TLS. Você também pode usar condições para conceder acesso a ações de serviço, mas somente se elas forem usadas por meio de um AWS service (Serviço da AWS) específico, como o CloudFormation. Para obter mais informações, consulte Elementos de política JSON do IAM: Condition.

Verifique o acesso entre contas e público aos recursos com o IAM Access Analyzer

Antes de conceder permissões para o acesso entre contas ou público na AWS, recomendamos que você verifique se esse acesso é necessário. É possível usar o IAM Access Analyzer para auxiliar na visualização e análise do acesso entre contas ou público para os tipos de recursos compatíveis. Você faz isso revisando as findings (descobertas) que o IAM Access Analyzer gera. Essas descobertas ajudam a verificar se seus controles de acesso a recursos concedem o acesso esperado. Além disso, ao atualizar as permissões entre contas e públicas, você pode verificar o efeito de suas alterações antes de implantar novos controles de acesso aos seus recursos. O IAM Access Analyzer também monitora continuamente os tipos de recursos compatíveis e gera uma descoberta de recursos que permitem o acesso entre contas ou público. Para obter mais informações, consulte Pré-visualização de acesso com APIs do IAM Access Analyzer.

Use o analisador de acesso do IAM para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais

Valide as políticas que você cria para garantir que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. É possível validar suas políticas usando a validação de política do IAM Access Analyzer. O IAM Access Analyzer oferece mais de 100 verificações de política e recomendações acionáveis para ajudar você a criar políticas seguras e funcionais. À medida que você cria novas políticas ou edita políticas existentes no console, o IAM Access Analyzer fornece recomendações para ajudar a aperfeiçoar e validar suas políticas antes de salvá-las. Além disso, recomendamos que você revise e valide todas as políticas existentes. Para obter mais informações, consulte IAM Access Analyzer policy validation (Validação de políticas do IAM Access Analyzer). Para obter mais informações sobre as verificações de política fornecidas pelo IAM Access Analyzer, consulte IAM Access Analyzer policy check reference (Referência de verificação de política do IAM Access Analyzer).

Estabeleça barreiras de proteção para permissões em várias contas

Conforme você escala suas workloads, separe-as usando várias contas gerenciadas com o AWS Organizations. Recomendamos que você use as políticas de controle de serviços (SCPs) do AWS Organizations para estabelecer barreiras de proteção para permissões a fim de controlar o acesso para todas as entidades principais (usuários e perfis do IAM) em todas as suas contas. Recomendamos que você use as políticas de controle de recursos (RCPs) do AWS Organizations para estabelecer barreiras de proteção para permissões a fim de controlar o acesso para recursos da

AWS em sua organização. As SCPs e as RCPs são tipos de políticas de organização que podem ser usadas para gerenciar permissões em sua organização no nível de organização da AWS, unidade organizacional (UO) ou conta.

No entanto, as SCPs e as RCPs, por si só, são insuficientes para conceder permissões a entidades principais e recursos em sua organização. Nenhuma permissão é concedida por SCPs e RCPs. Para conceder permissões, você deve anexar políticas baseadas em identidade ou em recursos para usuários do IAM, perfis do IAM ou recursos em suas contas. Para obter mais informações, consulte Componentes básicos do SRA: AWS Organizations, contas e barreiras de proteção.

Use limites de permissões para delegar o gerenciamento de permissões em uma conta

Em alguns cenários, você pode desejar delegar o gerenciamento de permissões em uma conta para outras pessoas. Por exemplo, você pode permitir que os desenvolvedores criem e gerenciem funções para suas workloads. Ao delegar permissões a outras pessoas, use limites de permissões para definir o máximo de permissões que você delega. Um limite de permissões é um recurso avançado que utiliza uma política gerenciada para definir o máximo de permissões que uma política baseada em identidade pode conceder a um perfil do IAM. Um limite de permissões não concede permissões por si só. Para obter mais informações, consulte Limites de permissões para entidades do IAM.

As práticas recomendadas do usuário-raiz para o Conta da AWS

Ao criar um pela primeira vez um Conta da AWS, você começa com um conjunto padrão de credenciais com acesso completo a todos os AWS recursos na sua conta. Essa identidade é chamada deConta da AWS usuário-raiz. É altamente recomendável que você não acesse o usuário-raiz Conta da AWS, a menos que tenha uma tarefa que exija credenciais de usuário-raiz. Você precisa proteger suas credenciais de usuário-raiz e seus mecanismos de recuperação de conta para certificar-se de que não exponha suas credenciais altamente privilegiadas para uso não autorizado.

Para várias Contas da AWS gerenciadas por meio do AWS Organizations, recomendamos remover as credenciais de usuário-raiz das contas dos membros para ajudar a evitar o uso não autorizado. É possível remover a senha do usuário-raiz, chaves de acesso e certificados de assinatura e desativar e excluir a autenticação multifator (MFA). As contas dos membros não podem fazer login com o usuário-raiz nem realizar a recuperação da senha do usuário-raiz. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

Em vez de acessar o usuário-raiz, crie um usuário administrativo para as tarefas diárias.

- Se você tiver uma Conta da AWS nova, consulte Configurar a Conta da AWS.
- Para vários Contas da AWS gerenciados AWS Organizations, consulte Configurar o Conta da AWS acesso para um usuário administrativo do IAM Identity Center.

Com seu usuário administrativo, você pode criar identidades adicionais para usuários que precisam de acesso a recursos em seu Conta da AWS. É altamente recomendável que você exija que os usuários se autentiquem com credenciais temporárias ao acessar AWS.

- Para um único e autônomo Conta da AWS, use Perfis do IAM para criar identidades em sua conta com permissões específicas. Funções são destinados a serem assumidas por qualquer pessoa que precise delas. Além disso, um perfil não tem credenciais de longo prazo padrão associadas a ele, como senha ou chaves de acesso. Em vez disso, quando você assumir um perfil, ele fornecerá credenciais de segurança temporárias para sua sessão de perfil. Ao contrário dos perfis do IAM, Usuários do IAM têm credenciais de longo prazo, como senhas e chaves de acesso. Sempre que possível, as práticas recomendadas aconselham a depender de credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso.
- Para várias Contas da AWS gerenciadas por meio do AWS Organizations, use os usuários da força de trabalho do Centro de Identidade do IAM Com o IAM Identity Center, você pode gerenciar centralmente os usuários em todas as suas contas Contas da AWS e as permissões dentro dessas contas. Gerencie as identidades de usuário com o IAM Identity Center ou de um provedor de identidade externo. Para obter mais informações, consulte O que é o Centro de Identidade do AWS IAM no Guia do usuário do Centro de Identidade do AWS IAM.

Tópicos

- Proteja suas credenciais de usuário-raiz para evitar o uso não autorizado
- Use uma senha de usuário-raiz forte para ajudar a proteger o acesso
- Proteja o acesso do seu usuário-raiz com autenticação multifator (MFA)
- Não crie chaves de acesso para o usuário-raiz
- Utilize aprovação por múltiplas pessoas para o login do usuário-raiz sempre que possível
- Use um endereço de e-mail do grupo para as credenciais do usuário-raiz
- Restrinja o acesso aos mecanismos de recuperação de conta

- Proteja as credenciais de usuário-raiz da sua conta do AWS Organizations
- Monitore o acesso e o uso

Proteja suas credenciais de usuário-raiz para evitar o uso não autorizado

Proteja as credenciais do usuário-raiz e use-as somente para as tarefas que as exigem. Para ajudar a evitar o uso não autorizado, não compartilhe sua senha de usuário-raiz, MFA, chaves de acesso, pares de chaves do CloudFront ou certificados de assinatura com ninguém, exceto aqueles que tenham uma necessidade comercial estrita de acessar o usuário-raiz.

Não armazene a senha do usuário-raiz com ferramentas que dependem de Serviços da AWS em uma conta que é acessada usando a mesma senha. Se você perder ou esquecer sua senha de usuário-raiz, não poderá acessar essas ferramentas. Recomendamos que você priorize a resiliência e considere exigir que duas ou mais pessoas autorizem o acesso ao local de armazenamento. Acesso à senha ou a seu local de armazenamento deve ser registrado e monitorado.

Use uma senha de usuário-raiz forte para ajudar a proteger o acesso

Recomendamos utilizar uma senha forte e exclusiva. Ferramentas como gerenciadores de senhas com algoritmos de geração de senhas fortes podem ajudar você a atingir esses objetivos. AWS exige que a senha atenda às seguintes condições:

- Ter no mínimo 8 caracteres e no máximo 128 caracteres de extensão.
- Incluir no mínimo três dos seguintes tipos de caracteres: maiúsculas, minúsculas, números e os símbolos ! @ # $ % ^ & * () <> [] {} | _+-=.
- Não ser idêntica ao nome ou endereço de e-mail da sua Conta da AWS.

Para obter mais informações, consulte Alterar a senha para o Usuário raiz da conta da AWS.

Proteja o acesso do seu usuário-raiz com autenticação multifator (MFA)

Como um usuário-raiz pode executar ações privilegiadas, é crucial adicionar a MFA (autenticação de múltiplos fatores) para o usuário-raiz como um segundo fator de autenticação, além do endereço de e-mail e senha como credenciais de login. É possível registrar até oito dispositivos de MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu usuário-raiz Conta da AWS.

Recomendamos enfaticamente habilitar vários dispositivos de MFA para suas credenciais de usuário-raiz a fim de fornecer flexibilidade e resiliência adicionais à sua estratégia de segurança. Todos os tipos de Conta da AWS (contas autônomas, de gerenciamento e de membros) exigem que a MFA seja configurada para o usuário-raiz. Os usuários precisarão registrar a MFA em até 35 dias após a primeira tentativa de login, para acessar o Console de gerenciamento da AWS, se a MFA ainda não estiver habilitada.

- As chaves de segurança de hardware certificadas pela FIDO são oferecidas por fornecedores terceirizados. Para obter mais informações, consulte Habilitar uma chave de segurança FIDO para o usuário-raiz Conta da AWS.
- Um dispositivo de hardware que gera um código numérico de seis dígitos com base no algoritmo de senha de uso único com marcação temporal (TOTP). Para obter mais informações, consulte Habilitar um token TOTP de hardware para o usuário-raiz Conta da AWS.
- Uma aplicação de autenticador virtual que é executada em um telefone ou outro dispositivo e emula um dispositivo físico. Para obter mais informações, consulte Habilitar um dispositivo de MFA virtual para o usuário-raiz Conta da AWS.

Não crie chaves de acesso para o usuário-raiz

As chaves de acesso permitem executar comandos na interface de linha de AWS comando (AWS CLI) ou usar operações de API de um dos AWS SDKs. É altamente recomendável que você não crie pares de chaves de acesso para seu usuário-raiz porque o usuário-raiz tem acesso total a todos os recursos e Serviços da AWS na conta, incluindo informações de faturamento.

Como apenas algumas tarefas exigem o usuário-raiz e elas normalmente são executadas com pouca frequência, recomendamos entrar no Console de gerenciamento da AWS para realizar as tarefas de usuário-raiz. Para acesso programático, em vez de criar chaves de acesso para o usuário-raiz, use o comando `aws login` com suas credenciais raiz para se autenticar na AWS CLI e nos SDKs. Essa abordagem fornece credenciais temporárias e alternadas automaticamente que aprimoram sua postura de segurança e eliminam a necessidade de gerenciar chaves de acesso de longo prazo. Antes de criar chaves de acesso, analise as Alternativas para chaves de acesso de longo prazo.

Utilize aprovação por múltiplas pessoas para o login do usuário-raiz sempre que possível

Considere usar a aprovação de várias pessoas para certificar-se de que nenhuma pessoa possa acessar o MFA e a senha do usuário-raiz. Algumas empresas adicionam uma camada adicional

de segurança configurando um grupo de administradores com acesso à senha e outro grupo de administradores com acesso à MFA. Um membro de cada grupo deve se reunir para fazer login como usuário-raiz.

Use um endereço de e-mail do grupo para as credenciais do usuário-raiz

Utilize um endereço de e-mail que seja gerenciado pela sua empresa e encaminhe as mensagens recebidas diretamente para um grupo de usuários. Se AWS precisar entrar em contato com o proprietário da conta, essa abordagem reduz o risco de atrasos na resposta, mesmo que as pessoas estejam de férias, afastadas por doença ou tenham deixado a empresa. O endereço de e-mail usado para o usuário-raiz não deve ser usado para outras finalidades.

Restrinja o acesso aos mecanismos de recuperação de conta

Certifique-se de desenvolver um processo para gerenciar os mecanismos de recuperação de credenciais do usuário-raiz no caso de precisar de acesso a eles durante uma situação de emergência, como a tomada de controle de sua conta administrativa.

- Certifique-se de ter acesso à caixa de entrada de e-mail do usuário-raiz para que você possa redefinir a senha perdida ou esquecida do usuário-raiz.
- Se a MFA do usuário-raiz da sua Conta da AWS estiver perdida, danificada ou não estiver funcionando, você poderá fazer login usando outra MFA registrada nas mesmas credenciais de usuário-raiz. Se você perdeu o acesso a todos os seus MFAs, você precisa que o número de telefone e o e-mail usados para registrar sua conta estejam atualizados e acessíveis para recuperar seu MFA. Para obter detalhes, consulte Recuperando um dispositivo de MFA de usuário-raiz.
- Se você optar por não armazenar a senha e a autenticação multifator (MFA) do usuário-raiz, então o número de telefone registrado em sua conta pode ser usado como uma alternativa para recuperar as credenciais do usuário-raiz. Certifique-se de ter acesso ao número de telefone de contato, mantenha o número de telefone atualizado e limite quem tem acesso para gerenciar o número de telefone.

Nenhuma pessoa deve ter acesso tanto à caixa de entrada de e-mail quanto ao número de telefone, uma vez que ambos são canais de verificação para recuperar a senha do usuário-raiz. É importante ter dois grupos de pessoas gerenciando esses canais. Um grupo terá acesso ao seu endereço de e-mail principal, enquanto outro grupo terá acesso ao número de telefone principal para recuperar o acesso à sua conta como usuário-raiz.

Proteja as credenciais de usuário-raiz da sua conta do AWS Organizations

Conforme você migra para uma estratégia de várias contas com o AWS Organizations, cada uma das suas Contas da AWS tem suas próprias credenciais de usuário-raiz que você precisa proteger. A conta que você usa para criar sua organização é a conta de gerenciamento e o resto das contas em sua organização são contas-membro.

Credenciais seguras de usuários-raiz para a conta de gerenciamento

AWSA exige que você registre a MFA para o usuário-raiz da conta de gerenciamento da sua organização. O registro da MFA deve ser concluído durante a primeira tentativa de login ou dentro do período de carência de 35 dias. Se a MFA não for habilitada dentro desse período, o registro será exigido antes que você possa acessar o Console de gerenciamento da AWS. Para obter mais informações, consulte Autenticação multifator para Usuário raiz da conta da AWS.

Credenciais seguras de usuário-raiz para contas-membro

Se você usa o AWS Organizations para gerenciar várias contas, há duas estratégias que pode adotar para proteger o acesso do usuário-raiz em seu AWS Organizations.

- Centralize o acesso ao usuário-raiz e remova as credenciais do usuário-raiz das contas dos membros. Remova as credenciais do usuário-raiz, as chaves de acesso e os certificados de assinatura e desative e exclua a autenticação multifator (MFA). Quando essa estratégia é usada, as contas de membros não podem fazer login com o usuário-raiz nem realizar a recuperação da senha do usuário-raiz. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.
- Proteja as credenciais do usuário-raiz das contas do AWS Organizations com a MFA para aumentar a segurança da conta. Para obter mais informações, consulte Autenticação multifator para Usuário raiz da conta da AWS.

Para obter detalhes, consulte Acessar contas de membro em sua organização no Guia do usuário do AWS Organizations.

Defina os controles de segurança preventivos no AWS Organizations usando uma política de controle de serviços (SCP)

Se as contas-membro na sua organização tiverem credenciais de usuário-raiz habilitadas, é possível aplicar uma SCP para restringir o acesso ao usuário-raiz da conta-membro. Negar todas as ações do usuário-raiz em suas contas-membro, exceto por ações específicas que são exclusivas da raiz, ajuda

a prevenir acessos não autorizados. Para obter detalhes, consulte Use um SCP para restringir o que o usuário-raiz de suas contas-membro pode fazer.

Monitore o acesso e o uso

Recomendamos que você use seus mecanismos de rastreamento atuais para monitorar, alertar e relatar o login e o uso das credenciais do usuário-raiz, incluindo alertas que anunciam o login e o uso do usuário-raiz. Os seguintes serviços podem ajudar a certificar-se de que o uso das credenciais do usuário-raiz seja rastreado e realizar verificações de segurança que podem ajudar a prevenir o uso não autorizado.

> **Note**
>
> O CloudTrail registra em log diferentes eventos de login para as sessões do usuário-raiz e do usuário-raiz privilegiado. Essas sessões privilegiadas permitem que tarefas que precisem de credenciais de usuário-raiz sejam realizadas em contas-membro na sua organização. É possível usar o evento de login para identificar as ações tomadas pela conta de gerenciamento ou por um administrador delegado usando `sts:AssumeRoot`. Para obter mais informações, consulte Rastreamento de tarefas privilegiadas no CloudTrail.

- Se você deseja ser notificado sobre a atividade de login do usuário-raiz em sua conta, pode aproveitar o Amazon CloudWatch para criar uma regra do Events que detecta quando as credenciais do usuário-raiz são usadas e aciona uma notificação para o administrador de segurança. Para obter detalhes, consulte Monitorar e notificar atividade de usuário-raiz Conta da AWS.
- Se quiser configurar notificações para alertá-lo sobre ações aprovadas do usuário-raiz, você pode utilizar o Amazon EventBridge junto com o Amazon SNS para escrever uma regra do EventBridge para rastrear o uso do usuário-raiz para a ação específica e notificá-lo usando um tópico do Amazon SNS. Por exemplo, consulte Enviar uma notificação quando um objeto do Amazon S3 é criado.
- Se você já está usando o GuardDuty como seu serviço de detecção de ameaças, você pode ampliar sua capacidade para notificá-lo quando as credenciais do usuário-raiz estão sendo usadas em sua conta.

Os alertas devem incluir, mas não se limitar a, o endereço de e-mail do usuário-raiz. Tenha procedimentos estabelecidos sobre como responder aos alertas, para que os funcionários que

recebem um alerta de acesso do usuário-raiz compreendam como validar se o acesso do usuário-raiz é esperado e saibam como proceder caso acreditem que um incidente de segurança está em andamento. Para um exemplo de como configurar alertas, consulte Monitorar e notificar a atividade do usuário-raiz Conta da AWS.

Avalie a conformidade do MFA do usuário-raiz

Os serviços a seguir podem ajudar a avaliar a conformidade com a MFA para as credenciais de usuário-raiz.

> As regras relacionadas à MFA retornarão não conformidade se você seguir a prática recomendada de remover as credenciais de usuário-raiz.
>
> Recomendamos remover as credenciais de usuário-raiz das contas dos membros na sua organização para ajudar a evitar o uso não autorizado. Depois de remover as credenciais do usuário-raiz, incluindo a MFA, essas contas de membros serão avaliadas como não aplicáveis.

- AWS ConfigO fornece regras para monitorar a conformidade com práticas recomendadas de usuário-raiz. É possível usar regras gerenciadas pelo AWS Config para ajudar a impor a MFA para as credenciais de usuário-raiz. O AWS Config também pode identificar chaves de acesso para o usuário-raiz.
- O Security Hub CSPM fornece uma visão abrangente do estado de segurança na AWS e ajuda a avaliar o ambiente da AWS em relação aos padrões e às práticas recomendadas do setor de segurança, como ter a MFA no usuário-raiz e não ter as chaves de acesso do usuário-raiz. Para obter detalhes sobre as regras disponíveis, consulte Controles do AWS Identity and Access Management no Guia do usuário do Security Hub CSPM.
- Trusted Advisor fornece uma verificação de segurança para que você saiba se o MFA não está habilitado na conta do usuário-raiz. Para obter mais informações, consulte MFA na conta raiz no Guia de suporte do usuárioAWS.

Se você precisar denunciar um problema de segurança em sua conta, consulte Denunciar e-mails suspeitos ou Relatar vulnerabilidades. Como alternativa, você pode Entrar em contato com a AWS para obter assistência e orientação adicional.

Casos de uso de negócios do IAM

Um caso de uso de negócios simples do IAM pode ajudar você a entender as maneiras básicas de implementar o serviço para controlar o acesso que seus usuários têm à AWS. O caso de uso é descrito em termos gerais, sem a mecânica de como você pode usaria a API do IAM para obter os resultados desejados.

Este caso de uso mostra duas formas típicas que uma empresa fictícia chamada Exemplo Corp pode usar o IAM. O primeiro cenário considera o Amazon Elastic Compute Cloud (Amazon EC2). O segundo considera o Amazon Simple Storage Service (Amazon S3).

Para obter mais informações sobre como usar o IAM com outros produtos da AWS, consulte AWSServiços da que funcionam com o IAM.

Tópicos

- Configuração inicial da Exemplo Corp
- Caso de uso do IAM com o Amazon EC2
- Caso de uso do IAM com o Amazon S3

Configuração inicial da Exemplo Corp

Nikki Wolf e Mateo Jackson são os fundadores da Example Corp. No início da empresa, eles criaram uma Conta da AWS e configuraram o Centro de Identidade do AWS IAM (Centro de Identidade do IAM) para criar contas administrativas para usar com seus recursos da AWS. Quando você configura o acesso à conta para o usuário administrativo, o IAM Identity Center cria um perfil do IAM correspondente. Esse perfil, que é controlado pelo Centro de Identidade do IAM, é criado na Conta da AWS relevante, e as políticas especificadas no conjunto de permissões AdministratorAccess são anexadas ao perfil.

Como agora têm contas de administrador, Nikki e Mateo não precisam mais usar o usuário raiz para acessar sua Conta da AWS. Pretendem usar apenas o usuário raiz para concluir as tarefas que somente o usuário raiz pode executar. Depois de analisar as práticas recomendadas de segurança, configuraram a autenticação multifator (MFA) da conta para credenciais do usuário raiz e decidiram proteger as credenciais do usuário raiz.

À medida que a empresa cresce, ele contrata funcionários para trabalhar como desenvolvedores, administradores, testadores, gerentes e administradores de sistema. Nikki é responsável por

operações, enquanto Mateo gerencia as equipes de engenharia. Eles configuraram um servidor de domínio do Active Directory para gerenciar as contas dos funcionários e o acesso aos recursos internos da empresa.

Para conceder aos funcionários acesso aos recursos da AWS, usam o Centro de Identidade do IAM para conectar o Active Directory da empresa à Conta da AWS.

Por terem conectado o Active Directory ao Centro de Identidade do IAM, os usuários, grupos e membros do grupo estão sincronizados e definidos. Eles devem atribuir conjuntos de permissões e perfis aos diferentes grupos para conceder aos usuários o nível correto de acesso aos recursos da AWS. Eles usam AWSPolíticas gerenciadas pela para funções de trabalho no Console de gerenciamento da AWS para criar esses conjuntos de permissões:

- Administrador
- Faturamento da
- Desenvolvedores
- Administradores de rede
- Administradores de banco de dados
- Administradores de sistemas
- Usuários de suporte

Em seguida, atribuem esses conjuntos de permissões aos perfis atribuídos aos grupos do Active Directory.

Para obter um guia detalhado que descreve a configuração inicial do Centro de Identidade do IAM, consulte Getting started (Conceitos básicos) no Guia do usuário do Centro de Identidade do AWS IAM. Para obter mais informações sobre como provisionar o acesso de usuários do Centro de Identidade do IAM, consulte Acesso via login único a contas da AWS, no Guia do usuário do Centro de Identidade do AWS IAM.

Caso de uso do IAM com o Amazon EC2

Uma empresa como a Exemplo Corp normalmente usa o IAM para interagir com serviços como o Amazon EC2. Para entender essa parte do caso de uso, você precisa de uma compreensão básica do Amazon EC2. Para obter mais informações sobre o Amazon EC2, consulte o Guia do usuário do Amazon EC2.

Permissões do Amazon EC2 para os grupos de usuários

Para fornecer controle de “perímetro”, Nikki anexa uma política ao grupo de usuários AllUsers. Esta política nega qualquer solicitação da AWS de um usuário se o endereço IP de origem estiver fora da rede corporativa da Exemplo Corp.

Na Exemplo Corp, diferentes grupos do IAM requerem diferentes permissões:

- Administradores de sistema: precisam de permissão para criar e gerenciar AMIs, instâncias, snapshots, volumes, grupos de segurança e assim por diante. Nikki anexa a política `AmazonEC2FullAccess` gerenciada pela AWS ao grupo de usuários SysAdmins que concede aos membros do grupo permissão para usar todas as ações do Amazon EC2.
- Desenvolvedores: precisam da capacidade de trabalhar apenas com instâncias. Mateo cria e anexa uma política ao grupo de usuários Developers (Desenvolvedores) que permite que os desenvolvedores chamem `DescribeInstances`, `RunInstances`, `StopInstances`, `StartInstances` e `TerminateInstances`.

  > **Note**
  >
  > O Amazon EC2 usa chaves SSH, senhas do Windows e grupos de segurança para controlar quem tem acesso ao sistema operacional de instâncias específicas do Amazon EC2. Não há um método no sistema do IAM para permitir ou negar acesso ao sistema operacional de uma instância específica.

- Usuários de suporte: não devem ser capazes de realizar qualquer ação do Amazon EC2, exceto listar os recursos do Amazon EC2 disponíveis atualmente. Portanto, Nikki cria e anexa uma política ao grupo de usuários Support (Suporte) que apenas permite que chamem as operações de API “Describe” do Amazon EC2.

Para obter exemplos práticos dessas políticas, consulte Exemplos de políticas baseadas em identidade do IAM e Uso do AWS Identity and Access Management no Guia do usuário do Amazon EC2.

Mudança da função de trabalho do usuário

Em determinado momento, um dos desenvolvedores, Paulo Santos, muda de cargo e se torna gerente. Como gerente, Paulo torna-se parte do grupo de usuários do Support e pode abrir casos de suporte para seus desenvolvedores. Mateo muda Paulo do grupo de usuários Developers

(Desenvolvedores) para o grupo de usuários Support (Suporte). Como resultado dessa mudança, a capacidade dele de interagir com instâncias do Amazon EC2 é limitada. Ele não pode executar ou iniciar instâncias. Ele também não pode interromper ou encerrar instâncias existentes, mesmo que ele tenha sido o usuário que executou ou iniciou a instância. Ele pode listar apenas as instâncias que os usuários da Exemplo Corp executaram.

Caso de uso do IAM com o Amazon S3

Empresas como a Exemplo Corp também costumam usar o IAM com o Amazon S3. João criou um bucket do Amazon S3 para a empresa, denominado amzn-s3-demo-bucket.

Criação de outros usuários e grupos de usuários

Como funcionários, Zhang Wei e Maria Major precisam ser capazes de criar seus próprios dados no bucket da empresa. Eles também precisam ler e gravar dados compartilhados em que todos os desenvolvedores trabalham. Para isso, Mateo organiza logicamente os dados do amzn-s3-demo-bucket usando um esquema de prefixos de chave do Amazon S3, conforme mostrado na figura a seguir.

```
/amzn-s3-demo-bucket
    /home
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo divide `/amzn-s3-demo-bucket` em um conjunto de diretórios iniciais para cada funcionário e uma área compartilhada para grupos de desenvolvedores e gerentes.

Agora, Mateo cria um conjunto de políticas para atribuir permissões aos usuários e grupos de usuários:

- Acesso ao diretório home para Zhang: Mateo anexa uma política a Wei que permite a ele ler, gravar e listar qualquer objeto com o prefixo de chaves `/amzn-s3-demo-bucket/home/zhang/` do Amazon S3
- Acesso ao diretório base para Mary: Mateo anexa uma política à Mary que permite a ela ler, gravar e listar objetos com o prefixo de chaves `/amzn-s3-demo-bucket/home/major/` do Amazon S3

- Acesso ao diretório compartilhado para o grupo de usuários desenvolvedores: Mateo anexa uma política ao grupo de usuários que permite aos desenvolvedores ler, gravar e listar objetos em `/amzn-s3-demo-bucket/share/developers/`
- Acesso ao diretório compartilhado para o grupo de usuários gerentes: Mateo anexa uma política ao grupo de usuários que permite aos gerentes ler, gravar e listar objetos em `/amzn-s3-demo-bucket/share/managers/`

> **Note**
>
> O Amazon S3 não concede permissão automática ao usuário que cria um bucket ou objeto para executar outras ações naquele bucket ou objeto. Portanto, em suas políticas do IAM, você deve explicitamente conceder aos usuários permissão para usar os recursos do Amazon S3 que eles criam.

Para obter exemplos dessas políticas, consulte Controle de acesso no Guia do usuário do Amazon Simple Storage Service. Para obter informações sobre como as políticas são avaliadas no runtime, consulte Lógica da avaliação de política.

Mudança da função de trabalho do usuário

Em determinado momento, um dos desenvolvedores, Zhang Wei, muda de cargo e se torna gerente. Presumimos que ele não precise mais acessar os documentos no diretório `share/developers`. Mateo, como administrador, move Wei para o grupo de usuários `Managers` e o remove do grupo de usuários `Developers`. Com esta simples reatribuição, Wei obtém automaticamente todas as permissões concedidas ao grupo de usuários `Managers`, mas não poderá mais acessar os dados no diretório `share/developers`.

Integração com uma empresa de terceiros

Organizações frequentemente trabalham com empresas, consultores e prestadores de serviço parceiros. A Example Corp tem um parceiro chamado Widget Company, e uma funcionária da Widget Company chamada Shirley Rodriguez precisa colocar dados em um bucket para uso da Example Corp. Nikki cria um grupo de usuários chamado WidgetCo e um usuário chamado `Shirley` e adiciona Shirley ao grupo de usuários WidgetCo. Nikki também cria um bucket especial denominado amzn-s3-demo-bucket1 para Shirley usar.

Nikki atualiza as políticas existentes ou adiciona novas políticas, a fim de acomodar o parceiro Widget Company. Por exemplo, Nikki pode criar uma nova política que nega aos membros do grupo de usuários WidgetCo a capacidade de usar qualquer ação exceto gravação. Essa política será necessária apenas se houver uma política ampla que forneça a todos os usuários acesso a um amplo conjunto de ações do Amazon S3.

Tutoriais do IAM

Os tutoriais a seguir apresentam procedimentos completos para tarefas comuns do AWS Identity and Access Management (IAM). Eles foram projetados para um ambiente de tipo de laboratório, com exemplos fictícios de nomes de empresas, nomes de usuários e assim por diante. O objetivo é fornecer orientação geral. Eles não se destinam ao uso direto em um ambiente de produção sem uma revisão e adaptação cuidadosas para atender às necessidades exclusivas do ambiente da organização.

Tutoriais

- Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM
- Tutorial do IAM: Criar e anexar sua primeira política gerenciada pelo cliente
- Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas
- Tutorial do IAM: Permitir que os usuários gerenciem suas credenciais e configurações de MFA
- Tutorial do IAM: use um modelo do CloudFormation para criar um provedor de identidades (IdP) do SAML
- Tutorial do IAM: use um modelo do CloudFormation para criar um perfil do IAM federado por SAML
- Tutorial do IAM: use um modelo do CloudFormation para criar um provedor de identidades (IdP) do SAML e um perfil do IAM federado por SAML

Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM

> **Important**
>
> As práticas recomendadas do IAM aconselham exigir que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias em vez de usuários do IAM com credenciais de longo prazo. Recomendamos usar somente usuários do IAM para casos de uso específicos não compatíveis com usuários federados.

Este tutorial ensina a usar um perfil para delegar acesso a recursos em diferentes Contas da AWS, denominadas Destino e Origem. Você compartilha recursos em uma conta com usuários em outra

conta. Ao configurar o acesso entre contas dessa forma, você não precisa criar usuários individuais do IAM em cada conta. Além disso, os usuários não precisam sair de uma conta e fazer login em outra conta para acessar recursos em diferentes Contas da AWS. Depois de configurar a função, você verá como usar a função do Console de gerenciamento da AWS, da AWS CLI e da API.

Neste tutorial, a conta Destino gerencia os dados de aplicações acessados por diferentes aplicações e equipes. Em cada conta, você armazena as informações da aplicação em buckets do Amazon S3. Você gerencia os usuários do IAM na conta Origem, em que tem dois perfis de usuário do IAM: Desenvolvedores e Analistas. Desenvolvedores e analistas usam a conta Origem para gerar dados compartilhados por vários microsserviços. Ambos os perfis têm permissões para trabalhar na conta Origem e acessar os recursos ali existentes. Periodicamente, um desenvolvedor deve atualizar os dados compartilhados na conta Destino. Os desenvolvedores armazenam esses dados em um bucket do Amazon S3 denominado `amzn-s3-demo-bucket-shared-container`.

Ao final deste tutorial, você terá o seguinte:

- Usuários na conta Origem (a conta confiável) com permissão para assumir um determinado perfil na conta Destino.
- Um perfil na conta Destino (a conta confiável) com permissão para acessar um determinado bucket do Amazon S3.
- O bucket `amzn-s3-demo-bucket-shared-container` na conta Destino.

Os desenvolvedores podem usar o perfil no Console de gerenciamento da AWS para acessar o bucket `amzn-s3-demo-bucket-shared-container` na conta Destino. Eles também podem acessar o bucket usando chamadas de API autenticadas por credenciais temporárias fornecidas pelo perfil. As tentativas semelhantes de usar um perfil feitas por um analista falham.

Esse fluxo de trabalho tem três etapas básicas:

Para criar um perfil na conta Destino

Primeiro, você usa o Console de gerenciamento da AWS para estabelecer confiança entre a conta Destino (número de identificação 999999999999) e a conta Origem (número de identificação 111111111111). Você começa criando um perfil do IAM denominado UpdateData. Ao criar o perfil, você define a conta Origem como uma entidade confiável e especifica uma política de permissões que deixa os usuários confiáveis atualizarem o bucket `amzn-s3-demo-bucket-shared-container`.

Conceder acesso ao perfil

Nesta sessão, você modifica a política de perfil para negar aos analistas acesso ao perfil `UpdateData`. Porque os analistas têm acesso de PowerUser nesse cenário, e você deve negar explicitamente a capacidade de usar o perfil.

Teste o acesso alternando funções

Por fim, como desenvolvedor, você usa o perfil `UpdateData` para atualizar o bucket `amzn-s3-demo-bucket-shared-container` na conta Destino. Você verá como acessar a função por meio do console da AWS da AWS CLI e da API.

Considerações

Antes de usar perfis do IAM para delegar acesso aos recursos nas Contas da AWS, é importante levar em conta o seguinte:

- Você não pode alternar um perfil ao fazer login como Usuário raiz da conta da AWS.
- As funções do IAM e as políticas baseadas em recurso delegam o acesso entre contas em uma única partição. Por exemplo, suponha que você tenha uma conta no Oeste dos EUA (Norte da Califórnia) na partição `aws` padrão. Além disso, você tem uma conta na China (Pequim) na partição `aws-cn`. Você não pode usar uma política baseada em recurso do Amazon S3 em sua conta na China (Pequim) para permitir o acesso de usuários em sua conta `aws` padrão.
- Você pode usar o Centro de Identidade do AWS IAM para facilitar a autenticação única (SSO) de Contas da AWS externas (contas fora da sua AWS Organizations) usando Security Assertion Markup Language (SAML). Para obter detalhes, consulte Integrate external Contas da AWS into Centro de Identidade do AWS IAM for central access management with independent billing using SAML 2.0
- Você pode associar perfis a recursos da AWS, como instâncias do Amazon EC2 ou funções do AWS Lambda. Para obter detalhes, consulte Criar um perfil para delegar permissões a um serviço da AWS.
- Se você quiser que uma aplicação assuma um perfil em outra Conta da AWS, poderá usar o AWS SDK para assumir perfis entre contas. Para obter mais informações, consulte Authentication and access no AWS SDKs and Tools Reference Guide.
- Trocar de função usando o Console de gerenciamento da AWS só funciona com contas que não exijam um `ExternalId`. Por exemplo, vamos supor que você conceda acesso à sua conta a terceiros e exija um `ExternalId` em um elemento `Condition` em sua política de permissões.

Nesse caso, o terceiro pode acessar sua conta somente usando a API da AWS ou uma ferramenta de linha de comando. O terceiro não pode usar o console, pois ele deve fornecer um valor para `ExternalId`. Para obter mais informações sobre esse cenário, consulte Acesso às Contas da AWS de propriedade de terceiros e How to enable cross account access to the Console de gerenciamento da AWS no AWS Security Blog.

Pré-requisitos

Este tutorial pressupõe que você já tenha os seguintes itens configurados:

- Duas Contas da AWS distintas que você pode usar, uma para representar a conta Origem e outra para representar a conta Destino.
- Usuários e perfis na conta Origem criados e configurados da seguinte forma:

Título do cargo	Usuário	Permissões
Desenvolvedor	David	Ambos os usuários podem entrar e usar o Console de gerenciamento da AWS na conta Origem.
Analista	Jane	

- Você não precisa criar nenhum usuário na conta Destino.
- Um bucket do Amazon S3 criado na conta Destino. Você pode chamá-lo de `amzn-s3-demo-bucket-shared-container` neste tutorial, mas como os nomes dos buckets do S3 devem ser globalmente exclusivos, você deve usar um bucket com outro nome.

Para criar um perfil na conta Destino

É possível permitir que os usuários de uma Conta da AWS acessem recursos em outra Conta da AWS. Neste tutorial, faremos isso criando um perfil que define quem pode acessá-lo e quais permissões são concedidas aos usuários que alternam para ele.

Nesta etapa do tutorial, você cria o perfil na conta Destino e especifica a conta Origem como uma entidade confiável. Você também limita as permissões da função a acesso somente leitura e gravação para o bucket `amzn-s3-demo-bucket-shared-container`. Qualquer pessoa com permissão para usar a função pode ler e gravar no bucket `shared-container`.

Antes de poder criar um perfil, você precisa do ID de conta da Conta da AWS Origem. Cada Conta da AWS tem um identificador ID de conta exclusivo e ela atribuído.

Para obter o ID da Conta da AWS Origem

1. Faça login no Console de gerenciamento da AWS como administrador da conta Origem e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No console do IAM, escolha seu nome de usuário na barra de navegação no canto superior direito. Geralmente, ele é assim: ***username***@***account_ID_number_or_alias***.

   Nesse cenário, você pode usar o ID de conta 111111111111 para a conta Origem. No entanto, é necessário usar um ID de conta válido se você usar este cenário em seu ambiente de teste.

Para criar um perfil na conta Destino que possa ser usado pela conta Origem

1. Entre no Console de gerenciamento da AWS como administrador da conta Destino e abra o console do IAM.
2. Para criar a função, prepare a política gerenciada que define as permissões para os requisitos da função. Em uma etapa posterior, você anexará essa política à função.

   Você deseja definir o acesso de leitura e gravação ao bucket `amzn-s3-demo-bucket-shared-container`. Embora a AWS forneça algumas políticas gerenciadas do Amazon S3, não há uma que conceda acesso de leitura e gravação a um único bucket do Amazon S3. Você pode criar a sua própria política.

   No painel de navegação, escolha Policies (Políticas) e, em seguida, selecione Create policy (Criar política).
3. Escolha a guia JSON e copie o texto do documento de política JSON a seguir. Cole este texto na caixa de texto JSON, substituindo o ARN do recurso (`arn:aws:s3:::shared-container`) pelo verdadeiro ARN para o seu bucket do Amazon S3.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-shared-container/*"
    }
  ]
}
```

A ação `ListAllMyBuckets` concede permissão para listar todos os buckets de propriedade do remetente autenticado da solicitação. A permissão `ListBucket` concede aos usuários a habilidade de visualizar objetos no bucket `amzn-s3-demo-bucket-shared-container`. As permissões `GetObject`, `PutObject`, `DeleteObject` concedem aos usuários a capacidade de visualizar, atualizar e excluir o conteúdo no bucket `amzn-s3-demo-bucket-shared-container`.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

4. Na página Revisar e criar, digite **`read-write-app-bucket`** para o nome da política. Revise as permissões concedidas pela política e depois escolha Criar política para salvar seu trabalho.

   As novas políticas aparecem na lista de políticas gerenciadas.

5. No painel de navegação, escolha Funções e Criar função.

6. Selecione o tipo de perfil Uma Conta da AWS.
7. Em ID da conta, digite a ID da conta Origem.

   Este tutorial usa o ID da conta de exemplo **111111111111** para a conta Origem. Você deve usar um ID de conta válido. Se você usar um ID de conta inválido, como **111111111111**, o IAM não permitirá que você crie a nova função.

   Por enquanto, não é necessário exigir um ID externo ou exigir que os usuários tenham autenticação multifator (MFA) para assumirem a função. Deixe essas opções desmarcadas. Para obter mais informações, consulte AWSCódigo da autenticação multifator no IAM da.
8. Selecione Next: Permissions (Próximo: permissões) para definir as permissões associadas à função.
9. Marque a caixa de seleção ao lado da política que você criou anteriormente.

   > **Dica**
   >
   > Em Filter (Filtrar), selecione Customer managed (Gerenciado pelo cliente) para filtrar a lista, de modo que ela inclua apenas as políticas criadas por você. Isso oculta as políticas criadas pela AWS e torna muito mais fácil encontrar a que você necessita.

   Em seguida, escolha Próximo.
10. (Opcional) Adicione metadados à função anexando etiquetas como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
11. (Opcional) Em Descrição da função, insira uma descrição para a nova função.
12. Depois de revisar a função, escolha Criar função.

    A função `UpdateData` é exibida na lista de funções.

Agora você deve obter o nome do recurso da Amazon (ARN) da função, que é um identificador exclusivo para a função. Ao modificar o perfil de desenvolvedor na conta Origem, você especifica o ARN do perfil da conta Destino para conceder ou negar as permissões.

Para obter o ARN para UpdateData

1. No painel de navegação do console do IAM, escolha Perfis.
2. Na lista de funções, escolha `UpdateData`.
3. Na seção Resumo do painel de detalhes, copie o valor de ARN da função.

   A conta Destino tem o ID de conta 999999999999, portanto, o ARN do perfil é `arn:aws:iam::999999999999:role/UpdateData`. Certifique-se de fornecer o ID da Conta da AWS verdadeiro para a conta Destino.

Neste ponto, você estabeleceu confiança entre as contas Destino e Origem. Você fez isso criando um perfil na conta Destino que identifica a conta Origem como uma entidade principal confiável. Você também definiu o que os usuários que passarem para a função `UpdateData` podem fazer.

Em seguida, modifique as permissões para o perfil Desenvolvedor.

Conceder acesso ao perfil

Nesse momento, os analistas e os desenvolvedores têm permissões para gerenciar os dados na conta Origem. Use as seguintes etapas necessárias para adicionar permissões a fim de permitir alternar para a função.

Para modificar o perfil Desenvolvedores para permitir que eles alternem para o perfil UpdateData

1. Entre como administrador na conta Origem e abra o console do IAM.
2. Escolha Perfis e depois Desenvolvedores.
3. Escolha a guia Permissions (Permissões), Add permissions (Adicionar permissões) e Create inline policy (Criar política em linha).
4. Selecione a guia JSON.
5. Adicione a seguinte instrução de política para permitir a ação `AssumeRole` no perfil `UpdateData` na conta Destino. Certifique-se de alterar *`DESTINATION-ACCOUNT-ID`* no elemento `Resource` para o ID da Conta da AWS real da conta Destino.

   JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/UpdateData"
    }
}
```

O efeito `Allow` permite explicitamente que o grupo Desenvolvedores acesse o perfil `UpdateData` na conta Destino. Qualquer desenvolvedor que tente acessar a função é bem-sucedido.

6. Selecione Revisar política.
7. Digite um Nome, como, por exemplo, **`allow-assume-S3-role-in-destination`**.
8. Escolha Criar política.

Na maioria dos ambientes, talvez o procedimento a seguir não seja necessário. Se, no entanto, você usar permissões PowerUserAccess, alguns grupos já poderão alternar funções. O procedimento a seguir mostra como adicionar uma permissão `"Deny"` ao grupo Analistas para garantir que eles não possam assumir o perfil. Se esse procedimento não for necessário em seu ambiente, recomendamos que não o adicione. As permissões `"Deny"` fazem com que seja mais complicado de gerenciar e entender o estado geral das permissões. Use as permissões `"Deny"` somente quando não existir uma opção melhor.

Para modificar o perfil Analistas para negar permissão de assumir o perfil **`UpdateData`**

1. Escolha Perfis e depois Analistas.
2. Escolha a guia Permissions (Permissões), Add permissions (Adicionar permissões) e Create inline policy (Criar política em linha).
3. Selecione a guia JSON.
4. Adicione a instrução de política a seguir para negar a ação `AssumeRole` na função `UpdateData`. Certifique-se de alterar *`DESTINATION-ACCOUNT-ID`* no elemento `Resource` para o ID da Conta da AWS real da conta Destino.

   JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": {
        "Effect": "Deny",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/UpdateData"
    }
}
```

O efeito `Deny` nega explicitamente ao grupo Analistas acesso ao perfil `UpdateData` na conta Destino. Qualquer analista que tente acessar o perfil receberá uma mensagem de acesso negado.

5. Selecione Revisar política.
6. Digite um nome, como **`deny-assume-S3-role-in-destination`**.
7. Escolha Criar política.

O perfil Desenvolvedores agora tem permissões para usar o perfil `UpdateData` na conta Destino. O perfil Analistas é impedido de usar o perfil `UpdateData`.

Em seguida, você verá como David, um desenvolvedor, pode acessar o bucket `amzn-s3-demo-bucket-shared-container` na conta Destino. David pode acessar o bucket no Console de gerenciamento da AWS, na AWS CLI ou na API do AWS.

Teste o acesso alternando funções

Após concluir as duas primeiras etapas deste tutorial, você terá um perfil que concede acesso a um recurso na conta Destino. Você também terá um perfil na conta Origem com usuários que têm permissão para usar esse perfil. Esta etapa discute como testar a alternância para essa função no Console de gerenciamento da AWS, na AWS CLI e na API da AWS.

Para obter ajuda com os problemas comuns que você pode encontrar ao trabalhar com perfis do IAM, consulte Solucionar problemas de perfis do IAM.

Alternar funções (console)

Se David precisar atualizar os dados na conta Destino no Console de gerenciamento da AWS, ele poderá fazer isso usando Switch Role. Ele especifica o ID da conta ou o alias e o nome da função, e suas permissões mudam imediatamente para as permitidas pela função. Depois, ele poderá usar o console para trabalhar com o bucket `amzn-s3-demo-bucket-shared-container`, mas

não poderá trabalhar com nenhum outro recurso em Destino. Enquanto David usar o perfil, não poderá usar também seus privilégios de usuário avançado na conta Origem. Isso porque apenas um conjunto de permissões pode ser ativado por vez.

O IAM oferece duas maneiras para David acessar a página Switch Role (Alternar função):

- David recebe um link do administrador que aponta para uma configuração predefinida Switch Role (Trocar de função). O link é fornecido ao administrador na última página do assistente Criar função ou na página Resumo da função de uma função entre contas. Ao selecionar esse link, David acessa a página Alternar função com os campos ID da conta e Nome da função já preenchidos. Tudo o que David precisa fazer é escolher Trocar perfis.
- O administrador não envia o link no e-mail, mas, em vez disso, envia os valores do número de ID da conta e do Nome da função. David deve inserir manualmente os valores para tocar de função. Isso é ilustrado no procedimento a seguir.

Para assumir uma função

1. David inicia sessão no Console de gerenciamento da AWS usando seu usuário normal na conta Origem.
2. Ele escolhe o link que o administrador enviou por e-mail para ele. Esse link leva David à página Switch Role (Trocar de função) com as informações de ID ou alias da conta e nome da função já preenchidas.

   —ou—

   David escolhe os nomes (menu Identity [Identidade]) na barra de navegação e, depois, escolhe Switch Roles (Trocar de função).

   Se essa for a primeira vez que David tenta acessar a página Alternar função dessa forma, primeiramente ele entrará na página Switch Role (Alternar função) de primeiro acesso. Essa página fornece informações adicionais sobre como a alternância de perfis pode permitir aos usuários gerenciar os recursos entre Contas da AWS. David deve selecionar Switch Role (Alternar função) nessa página para concluir o restante do procedimento.
3. Em seguida, para acessar o perfil, David deve digitar manualmente o número do ID da conta Destino (`999999999999`) e o nome do perfil (`UpdateData`).

   Além disso, David deseja monitorar quais funções e permissões associadas estão ativas no IAM no momento. Para controlar essas informações, ele digita `Destination` na caixa de texto

Display Name (Nome de exibição), escolhe a opção na cor vermelha e escolhe Switch Role (Alternar função).

4. David agora pode usar o console do Amazon S3 para trabalhar com o bucket do Amazon S3 ou qualquer outro recurso ao qual a função `UpdateData` tenha permissões.
5. Quando concluído, David poderá voltar para as permissões originais. Para isso, ele escolhe o nome de exibição do perfil Destino na barra de navegação e depois escolhe Voltar para David em 111111111111.
6. Da próxima vez que David desejar alternar perfis e escolher o menu Identidade na barra de navegação, ele verá a entrada Destino no mesmo local em que estava da última vez. Ele pode simplesmente escolher essa entrada para alternar funções imediatamente sem inserir novamente o ID da conta e o nome da função.

Alternar funções (AWS CLI)

Se David precisar trabalhar no ambiente Destino na linha de comando, poderá fazer isso usando a AWS CLI. Ele executa o comando `aws sts assume-role` e transmite o nome de recurso da Amazon (ARN) para obter credenciais de segurança temporárias para essa função. Em seguida, ele configura essas credenciais em variáveis do ambiente para que os comandos da AWS CLI funcionem usando as permissões da função. Enquanto David estiver usando o perfil, não poderá usar também seus privilégios de usuário avançado na conta Origem, pois apenas um conjunto de permissões pode estar ativo de cada vez.

Todas as chaves de acesso e tokens são apenas exemplos e não podem ser usados da forma que são mostrados. Substitua pelos valores apropriados do seu ambiente real.

Para assumir uma função

1. David abre uma janela de prompt de comando e confirma que o cliente de AWS CLI está funcionando executando o comando:

```
aws help
```

> **Note**
>
> O ambiente padrão de David usa as credenciais do usuário `David` do seu perfil padrão que ele criou com o comando `aws configure`. Para obter mais informações, consulte

Configuração da AWS Command Line Interface no Guia do usuário da AWS Command Line Interface.

2. Ele inicia o processo de alternar perfis executando o comando a seguir para alternar para o perfil `UpdateData` na conta Destino. Ele recebeu o ARN de função do administrador que criou a função. O comando também exige que você forneça um nome de sessão, você pode escolher qualquer texto que desejar para isso.

```
aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateData" --role-
session-name "David-ProdUpdate"
```

Em seguida, David vê a seguinte na saída:

```
{
    "Credentials": {
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDy
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

3. David vê três elementos de que precisa na seção Credentials Credenciais) da saída.

   - `AccessKeyId`
   - `SecretAccessKey`
   - `SessionToken`

   David precisa configurar o ambiente de AWS CLI para usar esses parâmetros nas chamadas subsequentes. Para obter informações sobre as várias maneiras de configurar suas credenciais, consulte Configuração da AWS Command Line Interface. Você não pode usar o comando `aws`

`configure` porque ele não dá suporte à captura do token de sessão. Porém, você pode inserir manualmente as informações em um arquivo de configuração. Como essas são credenciais temporárias com um tempo de expiração relativamente curto, é mais fácil adicioná-las ao ambiente de sua sessão de linha de comando atual.

4. Para adicionar os três valores ao ambiente, David recorta e cola a saída da etapa anterior nos seguintes comandos. Talvez você queira recortar e colar em um editor de texto simples para resolver problemas de quebra de linha na saída do token de sessão. Ela deve ser adicionada como uma string longa simples, apesar de a linha ser mostrada quebrada aqui para fins de clareza.

   O exemplo a seguir mostra os comandos fornecidos no ambiente Windows, onde "set" é o comando para criar uma variável de ambiente. Nos computadores Linux ou macOS, o comando é "exportar". Todas as outras partes do exemplo são válidas em todos os três ambientes.

   Para obter detalhes sobre como usar as ferramentas para Windows Powershell, consulte Alternar para um perfil do IAM (Ferramentas para Windows PowerShell)

   ```
   set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
   set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
   +Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
   Ryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
   uZEXAMPLECihzFB5lTYLto9dyBgSDyEXA
   MPLEKEY9/
   g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
   EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
   +4eZScEXAMPLENhykxiHen
   DHq6ikBQ==
   ```

   Nesse ponto, todos os comandos a seguir são executados de acordo com as permissões de função identificadas por essas credenciais. No caso de David, a função `UpdateData`.

   > **Important**
   >
   > Você pode salvar as definições de configuração usadas com frequência e credenciais em arquivos que são mantidos pela AWS CLI. Para obter mais informações, consulte Using existing configuration and credentials files no AWS Command Line Interface User Guide.

5. Execute o comando para acessar os recursos na conta Destino. Neste exemplo, David lista o conteúdo do bucket do S3 com o comando a seguir.

   ```
   aws s3 ls s3://shared-container
   ```

   Como nomes de bucket do Amazon S3 são universalmente exclusivos, não há necessidade de especificar o ID da conta que tem o bucket. Para acessar recursos de outros serviços da AWS, consulte a documentação da AWS CLI desse serviço para obter os comandos e a sintaxe necessários para referenciar seus recursos.

Uso da AssumeRole (API da AWS)

Quando David precisa fazer uma atualização na conta Destino no código, ele faz uma chamada `AssumeRole` para assumir o perfil `UpdateData`. A chamada retorna credenciais temporárias que ele pode usar para acessar o bucket `amzn-s3-demo-bucket-shared-container` na conta Destino. Com essas credenciais, David pode fazer chamadas de API para atualizar o bucket `amzn-s3-demo-bucket-shared-container`. Porém, ele não pode fazer chamadas de API para acessar nenhum outro recurso na conta Destino, mesmo tendo permissões de usuário avançado na conta Origem.

Para assumir uma função

1. David chama `AssumeRole` como parte de um aplicativo. Ele deve especificar o ARN `UpdateData`: `arn:aws:iam::999999999999:role/UpdateData`.

   A resposta da chamada `AssumeRole` inclui as credenciais temporárias com um `AccessKeyId` e um `SecretAccessKey`. Também inclui um horário de `Expiration` que indica quando as credenciais expiram, e é necessário solicitar novas. Quando você configura o encadeamento de perfis com o AWS SDK, muitos provedores de credenciais atualizam automaticamente as credenciais antes que expirem.
2. Com as credenciais temporárias, David faz uma chamada `s3:PutObject` para atualizar o bucket `amzn-s3-demo-bucket-shared-container`. Ele pode passar as credenciais para a chamada de API como o parâmetro `AuthParams`. Como as credenciais temporárias do perfil têm apenas acesso de leitura e gravação ao bucket `amzn-s3-demo-bucket-shared-container`, todas as outras ações na conta Destino são negadas.

Para obter um exemplo de código (usando Python), consulte Alternar para um perfil do IAM (API da AWS).

Recursos adicionais

Os seguintes recursos podem ajudar você a saber mais sobre os tópicos deste tutorial:

- Para obter mais informações sobre os usuários do IAM, consulte Identidades do IAM.
- Para mais informações sobre os buckets do Amazon S3, consulte Criar um bucket no Guia do usuário do Amazon Simple Storage Service.
- Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

Resumo

Você concluiu o tutorial de acesso à API entre contas. Você criou uma função para estabelecer confiança com outra conta e definiu quais ações as entidades confiáveis podem executar. Depois, você modificou uma política de perfil para controlar quais usuários do IAM; podem acessar o perfil. Como resultado, da conta Origem os desenvolvedores podem fazer atualizações no bucket `amzn-s3-demo-bucket-shared-container` na conta Destino usando credenciais temporárias.

Tutorial do IAM: Criar e anexar sua primeira política gerenciada pelo cliente

Neste tutorial, você usará o Console de gerenciamento da AWS para criar uma política gerenciada pelo cliente e anexará essa política a um usuário do IAM na sua Conta da AWS. A política que você cria permite que um usuário de teste do IAM faça login diretamente no Console de gerenciamento da AWS com permissões somente leitura.

Esse fluxo de trabalho tem três etapas básicas:

Etapa 1: Criar a política

> Por padrão, os usuários do IAM não têm permissões para fazer nada. Eles não podem acessar o Console de Gerenciamento da AWS ou gerenciar os dados, a menos que você permita. Nesta etapa, você cria uma política gerenciada pelo cliente que permite a qualquer usuário anexado fazer login no console.

Etapa 2: Anexar a política

Quando você anexa uma política a um usuário, ele herda todas as permissões de acesso associadas a essa política. Nesta etapa, você anexa a nova política a um usuário de teste.

Etapa 3: Testar o acesso do usuário

Assim que a política é anexada, você pode fazer login como o usuário e testá-la.

Pré-requisitos

Para executar as etapas neste tutorial, você precisa já ter o seguinte:

- Uma Conta da AWS com a qual você possa fazer login como usuário do IAM com permissões administrativas.
- Um usuário de teste do IAM que não tenha permissões atribuídas ou associações de grupo da seguinte forma:

Nome de usuário	Grupo	Permissões
PolicyUser	<nenhum>	<nenhum>

Etapa 1: Criar a política

Nesta etapa, você cria uma política gerenciada pelo cliente que permita a qualquer usuário anexado fazer login no Console de gerenciamento da AWS com acesso somente leitura aos dados do IAM.

Para criar a política para o usuário de teste

1. Faça login no console do IAM em https://console.aws.amazon.com/iam/ com seu usuário que tenha permissões de administrador.
2. No painel de navegação, selecione Políticas.
3. No painel de conteúdo, escolha Criar política.
4. Escolha a opção JSON e copie o texto do documento da política JSON a seguir. Cole este texto na caixa de texto do JSON.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    } ]
}
```

5. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a validação de política e depois escolha Próximo.

> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. No entanto, se você fizer alterações ou escolher a opção Review policy (Revisar política) na guia Visual editor (Editor visual), o IAM poderá reestruturar sua política de forma a otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

6. Na página Revisar e criar, digite **`UsersReadOnlyAccessToIAMConsole`** para o nome da política. Revise as permissões concedidas pela política e depois escolha Criar política para salvar seu trabalho.

   A nova política aparece na lista de políticas gerenciadas e está pronta para ser anexada.

Etapa 2: Anexar a política

Em seguida, anexe a política que você acabou de criar ao usuário de teste do IAM.

Para anexar a política ao usuário de teste

1. No console do IAM, no painel de navegação, escolha Policies (Políticas).

2. Na parte superior da lista de políticas, na caixa de pesquisa, comece a digitar **`UsersReadOnlyAccessToIAMConsole`** até ver a política. Depois, escolha o botão de seleção ao lado de UsersReadOnlyAccessToIAMConsole na lista.
3. Selecione o botão Actions (Ações) e escolha Attach (Anexar).
4. Nas entidades do IAM, escolha a opção de filtrar por Usuários.
5. Na caixa de pesquisa, comece a digitar **`PolicyUser`** até que o usuário fique visível na lista. Depois, marque a caixa de seleção ao lado desse usuário na lista.
6. Escolha Anexar política.

Você anexou a política ao usuário de teste do IAM, o que significa que o usuário agora tem acesso somente leitura ao console do IAM.

Etapa 3: Testar o acesso do usuário

Para este tutorial, recomendamos que você teste o acesso, fazendo login como o usuário de teste para ver o que os usuários podem experimentar.

Para testar o acesso fazendo login com seu usuário de teste

1. Faça login no console do IAM em https://console.aws.amazon.com/iam/ com o seu usuário de teste `PolicyUser`.
2. Navegue pelas páginas do console e tente criar um novo usuário ou grupo. Observe que `PolicyUser` pode exibir dados, mas não pode criar ou modificar dados existentes do IAM.

Recursos relacionados

Para obter informações relacionadas, consulte os recursos a seguir:

- Políticas gerenciadas e em linha
- Controlar o acesso do usuário do IAM ao Console de gerenciamento da AWS

Resumo

Você concluiu com êxito todas as etapas necessárias para criar e anexar uma política gerenciada pelo cliente. Como resultado, você pode fazer login no console do IAM com sua conta de teste para ver como é a experiência para os usuários.

Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define as permissões com base em atributos. Na AWS, esses atributos são chamados de tags. Você pode anexar etiquetas a recursos do IAM, incluindo entidades (usuários ou funções) do IAM, e a recursos da AWS. Você pode ainda definir políticas que usam chaves de condição de tag para conceder permissões aos seus principais com base nas tags. Ao usar tags para controlar o acesso aos recursos da AWS, você permite que suas equipes e recursos se expandam com menos alterações nas políticas da AWS. As políticas de ABAC são mais flexíveis do que as políticas tradicionais da AWS, nas quais é obrigatório listar cada recurso individual. Para obter mais informações sobre o ABAC e sua vantagem em relação às políticas tradicionais, consulte Definir permissões com base em atributos com autorização ABAC.

> **Note**
>
> Você deve passar um único valor para cada etiqueta de sessão. O AWS Security Token Service não oferece suporte a etiquetas de sessão de vários valores.

Tópicos

- Visão geral do tutorial
- Pré-requisitos
- Etapa 1: Criar usuários de teste
- Etapa 2: Criar a política de ABAC
- Etapa 3: Criar funções
- Etapa 4: Testar a criação de segredos
- Etapa 5: Testar a visualização de segredos
- Etapa 6: Testar a escalabilidade
- Etapa 7: Testar a atualização e a exclusão de segredos
- Resumo
- Recursos relacionados
- Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC

Visão geral do tutorial

Este tutorial mostra como criar e testar uma política que permite que funções do IAM com etiquetas de entidades acessem recursos com etiquetas correspondentes. Quando um principal faz uma solicitação para a AWS, suas permissões são concedidas com base na correspondência entre as tags de principal e de recurso. Essa estratégia permite que os indivíduos visualizem ou editem apenas os recursos da AWS necessários para seus trabalhos.

Cenário

Vamos supor que você seja um desenvolvedor líder em uma grande empresa chamada Example Corporation e seja um administrador experiente do IAM. Você está familiarizado com a criação e o gerenciamento de usuários, funções e políticas do IAM. Você quer garantir que os engenheiros de desenvolvimento e membros da equipe de controle de qualidade possam acessar os recursos necessários. Também é necessária uma estratégia que possa ser dimensionada à medida que sua empresa cresce.

Você escolhe usar as etiquetas de recursos da AWS e as etiquetas de entidades de função do IAM para implementar uma estratégia ABAC para serviços compatíveis com ela, começando com o AWS Secrets Manager. Para saber quais serviços oferecem suporte à autorização com base em tags, consulte AWSServiços da que funcionam com o IAM. Para saber quais chaves de condição de marcação você pode usar em uma política com as ações e recursos de cada serviço, consulte Ações, recursos e chaves de condição de serviços da AWS. Você pode configurar seu provedor de identidade da Web ou com base em SAML para passar tags de sessão para a AWS. Quando seus funcionários se agrupam na AWS, os atributos deles são aplicados ao principal resultante na AWS. Você pode usar o ABAC para conceder ou não permissões com base nesses atributos. Para saber como o uso de tags de sessão com uma identidade federada SAML difere deste tutorial, consulte Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC.

Os membros da equipe de engenharia e controle de qualidade estão no projeto Pegasus ou Unicorn . Escolha os seguintes valores de tag de projeto e equipe de 3 caracteres:

- `access-project` = `peg` para o projeto Pegasus
- `access-project` = `uni` para o projeto Unicorn
- `access-team` = `eng` para a equipe de engenharia
- `access-team` = `qas` para a equipe de controle de qualidade

Além disso, escolha se deseja exigir a tag de alocação de custos `cost-center` para habilitar relatórios personalizados de faturamento da AWS. Para obter mais informações, consulte Usar tags de alocação de custos no Guia do usuário do Gerenciamento de Faturamento e Custos da AWS.

Resumo das principais decisões

- Os funcionários fazem login com credenciais de usuário do IAM e assumem a função do IAM para suas respectivas equipes e projetos. Se sua empresa tiver seu próprio sistema de identidade, será possível configurar a federação para permitir que os funcionários assumam uma função sem usuários do IAM. Para obter mais informações, consulte Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC.
- A mesma política é anexada a todas as funções. As ações são permitidas ou negadas com base em tags.
- Os funcionários poderão criar novos recursos, mas somente se anexarem as mesmas tags ao recurso que são aplicadas à sua função. Isso garante que os funcionários possam visualizar o recurso depois de criá-lo. Os administradores não precisam mais atualizar políticas com o ARN de novos recursos.
- Os funcionários podem ler recursos de propriedade de sua equipe, independentemente do projeto.
- Os funcionários podem atualizar e excluir recursos de propriedade de sua própria equipe e do projeto.
- Os administradores do IAM podem adicionar uma nova função para novos projetos. Eles podem criar e etiquetar um novo usuário do IAM para permitir o acesso à função apropriada. Os administradores não precisam editar uma política para oferecer suporte a um novo projeto ou membro da equipe.

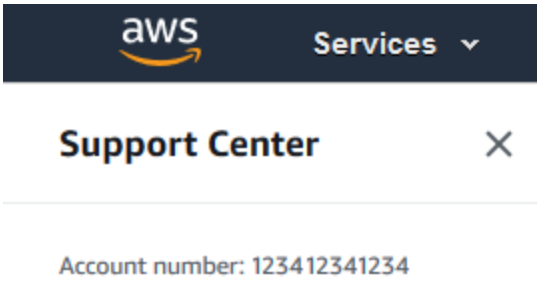
Neste tutorial, você marcará cada recurso, marcará suas funções de projeto e adicionará políticas às funções para permitir o comportamento descrito anteriormente. A política resultante permite que as funções `Create`, `Read`, `Update` e `Delete` acessem recursos marcados com as mesmas tags de projeto e equipe. A política também permite o acesso `Read` entre projetos para recursos marcados com a mesma equipe.
O diagrama mostra dois projetos em que os perfis estão limitados a acesso somente de leitura fora do projeto, embora tenham permissões para criar, ler, atualizar e excluir recursos em seu próprio projeto.

Pré-requisitos

Para executar as etapas neste tutorial, você já deve ter o seguinte:

- Uma Conta da AWS com a qual você possa fazer login como usuário com permissões administrativas.
- O ID de conta de 12 dígitos, que você usa para criar as funções na etapa 3.

  Para localizar o ID da conta da AWS usando o Console de gerenciamento da AWS, selecioneSuporte na barra de navegação do canto superior direito e selecione Support Center. O número da conta (ID) aparece no painel de navegação à esquerda.



- Experimente criar e editar usuários, funções e políticas do IAM no Console de gerenciamento da AWS. No entanto, se você precisar de ajuda para lembrar de um processo de gerenciamento do IAM, este tutorial fornece links por meio dos quais é possível visualizar instruções detalhadas.

Etapa 1: Criar usuários de teste

Para testes, crie quatro usuários do IAM com permissões para assumir funções com as mesmas etiquetas. Isso facilita a adição de mais usuários às suas equipes. Ao marcar os usuários, eles recebem acesso automaticamente para assumir a função correta. Não será necessário adicionar os usuários à política de confiança da função se eles trabalharem apenas em um projeto e equipe.

1. Crie a seguinte política gerenciada pelo cliente chamada `access-assume-role`. Para obter mais informações sobre como criar uma política JSON, consulte Criação de políticas do IAM.

   Política de ABAC: assumir qualquer função de ABAC, mas somente quando as tags da função e do usuário forem correspondentes

   A política a seguir permite que um usuário assuma qualquer função em sua conta com o prefixo de nome `access-`. A função também deve ser marcada com as mesmas tags de projeto, equipe e centro de custos que o usuário.

   Para usar esta política, substitua o texto do espaço reservado em itálico pelas informações da conta.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::111122223333:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/
access-project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/
access-team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

Para dimensionar este tutorial para um grande número de usuários, é possível anexar a política a um grupo e adicionar cada usuário ao grupo. Para obter mais informações, consulte Criar grupos do IAM e Editar usuários em grupos do IAM.

2. Crie os usuários do IAM a seguir e anexe a política de permissões `access-assume-role`. Certifique-se de selecionar Fornecer ao usuário acesso ao Console de gerenciamento da AWS e depois adicione as seguintes tags.

Nome de usuário	Chave de tag de usuário	Valor de tag de usuário
access-Arnav-peg-eng	`access-project`	`peg`
	`access-team`	`eng`

Nome de usuário	Chave de tag de usuário	Valor de tag de usuário
	`cost-center`	`987654`

access-Mary-peg-qas	`access-project`
	`access-team`
	`cost-center`
		`peg`
		`qas`
		`987654`

access-Saanvi-uni-eng	`access-project`
	`access-team`
	`cost-center`
		`uni`
		`eng`
		`123456`

access-Carlos-uni-qas	`access-project`
	`access-team`
	`cost-center`
		`uni`
		`qas`
		`123456`

Etapa 2: Criar a política de ABAC

Crie a política a seguir chamada **`access-same-project-team`**. Você adicionará essa política às funções em uma etapa posterior. Para obter mais informações sobre como criar uma política JSON, consulte Criação de políticas do IAM.

Para obter políticas adicionais que podem ser adaptadas para este tutorial, consulte as seguintes páginas:

- Controle de acesso de entidades de segurança do IAM
- Amazon EC2: permite iniciar ou interromper instâncias do EC2 que um usuário etiquetou, de forma programática e no console
- EC2: iniciar ou interromper instâncias baseadas em etiquetas de entidade de segurança e recurso correspondentes
- EC2: iniciar ou interromper instâncias baseadas em etiquetas
- IAM: assumir funções que têm uma etiqueta específica

Política de ABAC: acessar recursos do Secrets Manager somente quando as etiquetas de entidade e recurso forem correspondentes

A política a seguir permitirá que os principais criem, leiam, editem e excluam recursos, mas somente quando esses recursos forem marcados com os mesmos pares de chave/valor que o principal. Quando um principal cria um recurso, ele deve adicionar as tags `access-project`, `access-team` e `cost-center` e com valores correspondentes às tags do principal. A política também permite adicionar as tags opcionais `Name` ou `OwnedBy`.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
     {
         "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
         "Effect": "Allow",
         "Action": "secretsmanager:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
                 "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-
team}",
                 "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
             },
             "ForAllValues:StringEquals": {
                 "aws:TagKeys": [
                     "access-project",
                     "access-team",
                     "cost-center",
                     "Name",
                     "OwnedBy"
                 ]
             },
             "StringEqualsIfExists": {
                 "aws:RequestTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
             }
```

```
            }
        },
        {
            "Sid": "AllResourcesSecretsManagerNoTags",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ReadSecretsManagerSameTeam",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:Describe*",
                "secretsmanager:Get*",
                "secretsmanager:List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
                }
            }
        },
        {
            "Sid": "DenyUntagSecretsManagerReservedTags",
            "Effect": "Deny",
            "Action": "secretsmanager:UntagResource",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringLike": {
                    "aws:TagKeys": "access-*"
                }
            }
        },
        {
            "Sid": "DenyPermissionsManagement",
            "Effect": "Deny",
            "Action": "secretsmanager:*Policy",
            "Resource": "*"
        }
    ]
```

```
}
```

O que essa política faz?

- A declaração `AllActionsSecretsManagerSameProjectSameTeam` permitirá todas as ações desse serviço em todos os recursos relacionados, mas somente se as tags de recurso forem correspondentes às tags de principal. Ao adicionar `"Action": "secretsmanager:*"` à política, ela se expande à medida que o Secrets Manager o faz. Se o Secrets Manager adicionar uma nova operação de API, não será necessário adicionar essa ação à instrução. A declaração implementa o ABAC usando três blocos de condição. A solicitação será permitida somente se todos os três blocos retornarem true.
  - O primeiro bloco de condição dessa declaração retornará true se as chaves de tag especificadas estiverem presentes no recurso e seus valores forem correspondentes às tags de principal. Esse bloco retorna false para tags não correspondentes ou para ações que não oferecem suporte à marcação de recursos. Para saber quais ações não são permitidas por esse bloco, consulte Ações, recursos e chaves de condição do AWS Secrets Manager. Essa página mostra que as ações executadas no tipo de recurso Secret (Segredo) oferecem suporte à chave de condição `secretsmanager:ResourceTag/tag-key`. Algumas ações do Secrets Manager não oferecem suporte a esse tipo de recurso, incluindo `GetRandomPassword` e `ListSecrets`. É necessário criar declarações adicionais para permitir essas ações.
  - O segundo bloco de condição retornará true se cada chave de tag passada na solicitação for incluída na lista especificada. Isso é feito usando `ForAllValues` com o operador de condição `StringEquals`. Se nenhuma chave ou nenhum subconjunto do conjunto de chaves for passado, a condição retornará true. Isso permite operações `Get*` que não permitem passar tags na solicitação. Se o solicitante incluir uma chave de tag que não estiver na lista, a condição retornará false. Cada chave de tag passada na solicitação deve corresponder a um membro dessa lista. Para obter mais informações, consulte Operadores de conjunto para chaves de contexto de vários valores.
  - O terceiro bloco de condição retornará true se a solicitação oferecer suporte à passagem de tags, se todas as três tags estiverem presentes e se forem correspondentes aos valores de tag do principal. Esse bloco também retorna true se a solicitação não oferecer suporte à passagem de tags. Isso ocorre devido a `...IfExists` no operador de condição. O bloco retornará false se não houver nenhuma tag passada durante uma ação que ofereça suporte a ele ou se as chaves e os valores das tags não forem correspondentes.

- A declaração `AllResourcesSecretsManagerNoTags` permite as ações `GetRandomPassword` e `ListSecrets` que não são permitidas pela primeira declaração.
- A declaração `ReadSecretsManagerSameTeam` permitirá operações somente leitura se o principal estiver marcado com a mesma tag access-team que o recurso. Isso é permitido independentemente da tag do projeto ou do centro de custos.
- A declaração `DenyUntagSecretsManagerReservedTags` nega solicitações para remover tags com chaves que começam com "access-" do Secrets Manager. Essas tags são usadas para controlar o acesso aos recursos, portanto, a remoção de tags pode remover permissões.
- A declaração `DenyPermissionsManagement` nega o acesso para criar, editar ou excluir políticas baseadas em recursos do Secrets Manager. Essas políticas podem ser usadas para alterar as permissões do segredo.

> **Important**
>
> Essa política usa uma estratégia para permitir todas as ações de um serviço, mas nega explicitamente ações que alteram permissões. Negar uma ação substitui qualquer outra política que permita que o principal execute essa ação. Isso pode ter resultados indesejados. Como melhor prática, use negações explícitas somente quando não houver nenhuma circunstância que permita essa ação. Caso contrário, permita uma lista de ações individuais, e as ações indesejadas serão negadas por padrão.

Etapa 3: Criar funções

Crie as funções do IAM a seguir e anexe a política **`access-same-project-team`** que você criou na etapa anterior. Para obter mais informações sobre como criar funções do IAM, consulte Criar um perfil para conceder permissões a um usuário do IAM. Se você optar por usar a federação em vez de usuários e funções do IAM, consulte Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC.

Função de trabalho	Nome do perfil	Tags de função	Descrição de função
Projeto Pegasus de engenharia	access-peg-engineering	access-project = peg access-team = eng	Permite que os engenheiros leiam todos os recursos

Função de trabalho	Nome do perfil	Tags de função	Descrição de função
		cost-center = 987654	de engenharia e criem e gerenciem recursos de engenharia do Pegasus.
Projeto Pegasus de controle de qualidade	access-peg-quality-assurance	access-project = peg access-team = qas cost-center = 987654	Permite que a equipe de controle de qualidade leia todos os recursos de controle de qualidade e crie e gerencie todos os recursos de controle de qualidade do Pegasus.
Projeto Unicorn de engenharia	access-uni-engineering	access-project= uni access-team = eng cost-center = 123456	Permite que os engenheiros leiam todos os recursos de engenharia e criem e gerenciem recursos de engenharia do Unicorn.

Função de trabalho	Nome do perfil	Tags de função	Descrição de função
Projeto Unicorn de controle de qualidade	access-uni-quality-assurance	access-project = `uni` access-team = `qas` cost-center = `123456`	Permite que a equipe de controle de qualidade leia todos os recursos de controle de qualidade e crie e gerencie todos os recursos de controle de qualidade do Unicorn.

Etapa 4: Testar a criação de segredos

A política de permissões anexada às funções permite que os funcionários criem segredos. Isso será permitido somente se o segredo estiver marcado com seu projeto, equipe e centro de custos. Verifique se suas permissões estão funcionando conforme o esperado, fazendo login como seus usuários, assumindo a função correta e testando a atividade no Secrets Manager.

Como testar a criação de um segredo com e sem as tags necessárias

1. Na janela principal do navegador, permaneça conectado como usuário administrador para que seja possível revisar usuários, funções e políticas no IAM. Use uma janela incognito do navegador ou um navegador separado para seus testes. Lá, faça login como o usuário do IAM `access-Arnav-peg-eng` e abra o console do Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Tente alternar para a função `access-uni-engineering`.

   Ocorrerá falha nessa operação porque os valores de tag `access-project` e `cost-center` não são correspondentes para o usuário `access-Arnav-peg-eng` e a função `access-uni-engineering`.

Para obter mais informações sobre alternância de perfis no Console de gerenciamento da AWS, consulte Mudar de um usuário para um perfil do IAM (console)

3. Alterne para a função `access-peg-engineering`.
4. Armazene um novo segredo usando as seguintes informações. Para saber como armazenar um segredo, consulte Criar um segredo básico no Guia do usuário do AWS Secrets Manager.

> **Important**
>
> O Secrets Manager exibe alertas de que você não tem permissões para os serviços adicionais da AWS que funcionam com o Secrets Manager. Por exemplo, para criar credenciais para um banco de dados do Amazon RDS, é necessário ter permissão para descrever instâncias do RDS, clusters do RDS e clusters do Amazon Redshift. Você pode ignorar esses alertas, pois não vai usar esses serviços específicos da AWS neste tutorial.

1. Na seção Select secret type (Selecionar tipo de segredo), escolha Other type of secrets (Outros tipos de segredos). Nas duas caixas de texto, insira `test-access-key` e `test-access-secret`.
2. Insira `test-access-peg-eng` no campo Secret name (Nome do segredo).
3. Adicione diferentes combinações de tags da tabela a seguir e visualize o comportamento esperado.
4. Escolha Store (Armazenar) para tentar criar o segredo. Quando o armazenamento falhar, volte para as páginas anteriores do console do Secrets Manager e use o próximo conjunto de etiquetas da tabela a seguir. O último conjunto de tags é permitido e criará o segredo com êxito.

A tabela a seguir mostra as combinações de tags de ABAC para o perfil `test-access-peg-eng`.

access-project Valor da tag	**access-team** Valor da tag	**cost-center** Valor da tag	Tags adicionais	Comportamento esperado
(none)	(none)	(none)	(none)	Negado porque o valor da tag `access-project` não é correspondente ao valor da função de `peg`.
`uni`	`eng`	`987654`	(none)	Negado porque o valor da tag `access-project` não é correspondente ao valor da função de `peg`.
`peg`	`qas`	`987654`	(none)	Negado porque o valor da tag `access-team` não é correspondente ao valor da função de `eng`.
`peg`	`eng`	`123456`	(none)	Negado porque o valor da tag `cost-center` não é correspondente ao valor da função de `987654`.
`peg`	`eng`	`987654`	Proprietário= Jane	Negado porque a tag adicional `owner` não é permitida pela política, mesmo que todas as três tags necessárias estejam presentes e seus valores correspondam aos valores da função.

access-project Valor da tag	**access-team** Valor da tag	**cost-center** Valor da tag	Tags adicionais	Comportamento esperado
peg	eng	987654	Nome = Jane	Permitido porque todas as três tags necessárias estão presentes e seus valores são correspondentes aos valores da função. Você também tem permissão para incluir a tag opcional Name.

5. Saia e repita as três primeiras etapas deste procedimento para cada um dos valores de função e de tag a seguir. Na quarta etapa deste procedimento, teste qualquer conjunto de tags ausentes, tags opcionais, tags não permitidas e valores de tags inválidos que você escolher. Depois disso, use as tags necessárias para criar um segredo com as tags e o nome a seguir.

Nome de usuário	Nome do perfil	Nome de segredo	Tags de segredo
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654
access-Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = uni access-team = qas cost-center = 123456

Etapa 5: Testar a visualização de segredos

A política anexada a cada função permite que os funcionários visualizem todos os segredos marcados com o nome da equipe, independentemente do projeto. Verifique se suas permissões estão funcionando conforme o esperado testando suas funções no Secrets Manager.

Como testar a visualização de um segredo com e sem as tags necessárias

1. Faça login como um dos seguintes usuários do IAM:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
2. Alterne para a função correspondente:
   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-uni-quality-assurance`

   Para obter mais informações sobre como alternar funções no Console de gerenciamento da AWS, consulte Mudar de um usuário para um perfil do IAM (console).
3. No painel de navegação à esquerda, escolha o ícone do menu para expandir o menu e escolha Secrets (Segredos).
4. Você deve ver todos os quatro segredos na tabela, independentemente da sua função atual. Isso é esperado porque a política chamada `access-same-project-team` permite a ação `secretsmanager:ListSecrets` para todos os recursos.
5. Escolha o nome de um dos segredos.
6. Na página de detalhes do segredo, as tags da função determinam se é possível visualizar o conteúdo da página. Compare o nome da função com o nome do segredo. Se eles compartilharem o mesmo nome de equipe, as tags `access-team` serão correspondentes. Se elas não forem correspondentes, o acesso será negado.

   A tabela a seguir mostra o comportamento de visualização de segredo do ABAC para cada perfil.

Nome do perfil	Nome de segredo	Comportamento esperado
`access-peg-engineering`	`test-access-peg-eng`	Permitido
	test-access-peg-qas	Denied
	test-access-uni-eng	Permitido
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-peg-eng	Denied
	test-access-peg-qas	Permitido
	test-access-uni-eng	Denied
	test-access-uni-qas	Permitido
access-uni-engineering	test-access-peg-eng	Permitido
	test-access-peg-qas	Denied
	test-access-uni-eng	Permitido
	test-access-uni-qas	Denied
access-uni-quality-assurance	test-access-peg-eng	Denied
	test-access-peg-qas	Permitido
	test-access-uni-eng	Denied
	test-access-uni-qas	Permitido

7. Na trilha de navegação na parte superior da página, escolha Secrets (Segredos) para retornar à lista de segredos. Repita as etapas neste procedimento usando funções diferentes para testar se é possível visualizar cada um dos segredos.

Etapa 6: Testar a escalabilidade

Um motivo importante para usar o controle de acesso baseado em atributo (ABAC) em vez do controle de acesso baseado em função (RBAC) é a escalabilidade. À medida que sua empresa adiciona novos projetos, equipes ou pessoas à AWS, não é necessário atualizar suas políticas orientadas pelo ABAC. Por exemplo, vamos supor que a Example Company esteja financiando um novo projeto, o código chamado Centaur. Uma engenheira chamada Saanvi Sarkar será a engenheira-chefe da Centaur enquanto continua trabalhando no projeto Unicorn . Saanvi também revisará o trabalho para projeto Peg. Há também vários engenheiros recém-contratados, incluindo Nikhil Jayashankar, que trabalhará apenas no projeto Centaur .

Como adicionar o novo projeto à AWS

1. Faça login como usuário administrador do IAM e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Roles (Funções) e adicione uma função do IAM chamada `access-cen-engineering`. Anexe a política de permissões **`access-same-project-team`** ao perfil e adicione as seguintes tags de perfil:
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
3. No painel de navegação à esquerda, escolha Uses (Usuários).
4. Adicione um novo usuário denominado `access-Nikhil-cen-eng`, anexe a política denominada `access-assume-role` e adicione as seguintes tags de usuário.
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
5. Use os procedimentos em Etapa 4: Testar a criação de segredos e Etapa 5: Testar a visualização de segredos. Em outra janela do navegador, teste se Nikhil pode criar apenas segredos de engenharia do Centaur e se ele pode visualizar todos os segredos de engenharia.
6. Na janela principal do navegador em que você fez login como administrador, escolha o usuário `access-Saanvi-uni-eng`.
7. Na guia Permissions (Permissões), remova a política de permissões access-assume-role.

8. Adicione a seguinte política em linha chamada `access-assume-specific-roles`. Para obter mais informações sobre como adicionar uma política em linha a um usuário, consulte Para incorporar uma política em linha de um usuário ou uma função (console).

   Política de ABAC: assumir apenas funções específicas

   Essa política permite que Saanvi assuma os perfis de engenharia nos projetos Pegasus e Centaur. É necessário criar essa política personalizada porque o IAM não oferece suporte a etiquetas de vários valores. Não é possível marcar o usuário de Saanvi com `access-project` = `peg` e `access-project` = `cen`. Além disso, o modelo de autorização da AWS não pode ser correspondente a ambos os valores. Para obter mais informações, consulte Regras para etiquetar no IAM e no AWS STS. Em vez disso, é necessário especificar manualmente as duas funções que ela pode assumir.

   Para usar esta política, substitua o texto do espaço reservado em itálico pelas informações da conta.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::111122223333:role/access-peg-engineering",
                "arn:aws:iam::111122223333:role/access-cen-engineering"
            ]
        }
    ]
}
```

9. Use os procedimentos em Etapa 4: Testar a criação de segredos e Etapa 5: Testar a visualização de segredos. Em outra janela do navegador, verifique se Saanvi pode assumir ambas as funções. Verifique se ela pode criar segredos apenas para seu projeto, equipe e centro de custos, dependendo das tags da função. Verifique também que ela pode visualizar detalhes sobre todos os segredos de propriedade da equipe de engenharia, incluindo os que ela acabou de criar.

Etapa 7: Testar a atualização e a exclusão de segredos

A política `access-same-project-team` anexada às funções permite que os funcionários atualizem e excluam todos os segredos marcados com seu projeto, equipe e centro de custos. Verifique se suas permissões estão funcionando conforme o esperado testando suas funções no Secrets Manager.

Como testar a atualização e a exclusão de um segredo com e sem as tags necessárias

1. Faça login como um dos seguintes usuários do IAM:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
   - `access-Nikhil-cen-eng`
2. Alterne para a função correspondente:
   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-peg-quality-assurance`
   - `access-cen-engineering`

   Para obter mais informações sobre como alternar funções no Console de gerenciamento da AWS, consulte Mudar de um usuário para um perfil do IAM (console).
3. Para cada função, tente atualizar a descrição do segredo e tente excluir os segredos a seguir. Para obter mais informações, consulte Modificar um segredo e Excluir e restaurar um segredo no Guia do usuário do AWS Secrets Manager.

   A tabela a seguir mostra a atualização e exclusão de segredos de ABAC para cada perfil.

Nome do perfil	Nome de segredo	Comportamento esperado
access-peg-engineering	test-access-peg-eng	Permitido

Nome do perfil	Nome de segredo	Comportamento esperado
	test-access-uni-eng	Denied
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-peg-qas	Permitido
	test-access-uni-eng	Denied
access-uni-engineering	test-access-uni-eng	Permitido
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-uni-qas	Permitido

Resumo

Agora você concluiu com êxito todas as etapas necessárias para usar tags para o controle de acesso baseado em atributo (ABAC). Você aprendeu a definir uma estratégia de marcação. Você aplicou essa estratégia aos seus principais e recursos. Você criou e aplicou uma política que impõe a estratégia para o Secrets Manager. Você também aprendeu que o ABAC é dimensionado facilmente ao adicionar novos projetos e membros da equipe. Como resultado, é possível fazer login no console do IAM com suas funções de teste e experimentar como usar etiquetas para ABAC na AWS.

> **Note**
>
> Você adicionou políticas que permitem ações somente sob condições específicas. Se você aplicar uma política diferente para seus usuários ou funções que tenha permissões mais amplas, as ações podem não estar limitadas a exigir marcação. Por exemplo, se você conceder permissões administrativas completas a um usuário usando a política gerenciada `AdministratorAccess` da AWS, essas políticas não vão restringir esse acesso. Para obter mais informações sobre como as permissões são determinadas quando várias políticas estão envolvidas, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso.

Recursos relacionados

Para obter informações relacionadas, consulte os recursos a seguir:

- Definir permissões com base em atributos com autorização ABAC
- AWSChaves de contexto de condições globais da
- Criar um perfil para conceder permissões a um usuário do IAM
- Tags para recursos do AWS Identity and Access Management
- Controlar o acesso a recursos da AWS usando tags
- Mudar de um usuário para um perfil do IAM (console)
- Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC

Para saber como monitorar as etiquetas em sua conta, consulte Monitorar alterações de etiquetas em recursos da AWS com fluxos de trabalho sem servidor e o Amazon CloudWatch Events.

Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define as permissões com base em atributos. Na AWS, esses atributos são chamados de tags. Você pode anexar etiquetas a recursos do IAM, incluindo entidades (usuários ou funções) do IAM, e a recursos da AWS. Quando as entidades são usadas para fazer solicitações AWS, elas se tornam principais e incluem tags.

Você também pode passar tags de sessão ao assumir uma função ou federar um usuário. Depois disso, é possível definir políticas que usam chaves de condição de tag para conceder permissões aos seus principais com base nas tags. Ao usar tags para controlar o acesso aos recursos da AWS, você permite que suas equipes e recursos se expandam com menos alterações nas políticas da AWS. As políticas de ABAC são mais flexíveis do que as políticas tradicionais da AWS, nas quais é obrigatório listar cada recurso individual. Para obter mais informações sobre o ABAC e sua vantagem em relação às políticas tradicionais, consulte Definir permissões com base em atributos com autorização ABAC.

Se sua empresa usar um provedor de identidade (IdP) baseado em SAML para gerenciar identidades corporativas de usuário, você pode usar atributos SAML para controle de acesso granular na AWS. Os atributos podem incluir identificadores do centro de custo, endereços de e-mail do usuário, classificações de departamento e atribuições de projeto. Ao passar esses atributos como tags de sessão, você pode controlar o acesso à AWS com base nessas tags de sessão.

Para concluir o tutorial ABAC passando atributos SAML para o principal de sessão, conclua as tarefas em Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas, com as alterações incluídas neste tópico.

Pré-requisitos

Para executar as etapas para usar tags de sessão SAML para ABAC, você já deve ter o seguinte:

- Acesso a um IdP baseado em SAML onde você possa criar usuários de teste com atributos específicos.
- A capacidade de fazer login com um usuário que tenha permissões de administrador.
- Experimente criar e editar usuários, funções e políticas do IAM no Console de gerenciamento da AWS. No entanto, se você precisar de ajuda para lembrar de um processo de gerenciamento do IAM, o tutorial de ABAC fornece links por meio dos quais é possível visualizar instruções detalhadas.
- Experiência na configuração de um IdP baseado em SAML no IAM. Para visualizar mais detalhes e links da documentação detalhada do IAM, consulte Passar tags de sessão usando AssumeRoleWithSAML.

Etapa 1: Criar usuários de teste

Siga as instruções em Etapa 1: Criar usuários de teste. Como suas identidades são definidas em seu provedor, não é necessário adicionar usuários do IAM para os seus funcionários.

Etapa 2: Criar a política de ABAC

Siga as instruções na Etapa 2: Criar a política de ABAC para criar a política gerenciada especificada no IAM.

Etapa 3: Criar e configurar a função SAML

Ao usar o tutorial ABAC para SAML, é necessário executar etapas adicionais para criar a função, configurar o IdP SAML e habilitar o acesso ao Console de gerenciamento da AWS. Para obter mais informações, consulte Etapa 3: Criar funções.

Etapa 3A: Criar a função SAML

Crie uma única função que confie no seu provedor de identidade SAML e no usuário `test-session-tags` criado na etapa 1. O tutorial ABAC usa funções distintas com diferentes tags de

função. Como você está passando tags de sessão do seu IdP SAML, é preciso apenas uma função. Para saber como criar uma função baseada em SAML, consulte Criar um perfil para uma federação do SAML 2.0 (console).

Nomeie a função `access-session-tags`. Anexe a política de permissões `access-same-project-team` à função. Edite a política de confiança da função para usar a política a seguir. Para obter instruções detalhadas sobre como editar a relação de confiança de uma função, consulte Atualizar a política de confiança de um perfil.

A política de confiança da função a seguir permite que seu provedor de identidade SAML e o usuário `test-session-tags` assumam a função. Ao assumirem a função, eles devem passar as três tags de sessão especificadas. A ação `sts:TagSession` é necessária para permitir a passagem de tags de sessão.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://region-
code.signin.aws.amazon.com/saml"}
            }
        }
```

```
    ]
}
```

A declaração `AllowSamlIdentityAssumeRole` permite que os membros das equipes de Engenharia e Garantia de Qualidade assumam essa função ao se federarem no Example Corporation IdP da AWS. O provedor SAML `ExampleCorpProvider` é definido no IAM. O administrador já configurou a declaração do SAML para passar as três tags de sessão necessárias. A declaração pode passar tags adicionais, mas essas três devem estar presentes. Os atributos da identidade podem ter qualquer valor para as tags `cost-center` e `access-project`. No entanto, o valor do atributo `access-team` deve corresponder a `eng` ou `qas` para indicar que a identidade está na equipe de Engenharia ou Garantia de Qualidade.

Etapa 3B: Configurar o IdP SAML

Configure seu IdP SAML para passar os atributos `cost-center`, `access-project` e `access-team` como tags de sessão. Para obter mais informações, consulte Passar tags de sessão usando AssumeRoleWithSAML.

Para passar esses atributos como tags de sessão, inclua os seguintes elementos em sua declaração do SAML.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
  <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
  <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
  <AttributeValue>eng</AttributeValue>
</Attribute>
```

Etapa 3C: habilitar o acesso ao console

Habilite o acesso ao console para seus usuários federados do SAML. Para obter mais informações, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS.

Etapa 4: Testar a criação de segredos

Federe no Console de gerenciamento da AWS usando a função `access-session-tags`. Para obter mais informações, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS. Depois, siga as instruções Etapa 4: Testar a criação de segredos para criar segredos. Use identidades SAML diferentes com atributos para corresponder às tags indicadas no tutorial ABAC. Para obter mais informações, consulte Etapa 4: Testar a criação de segredos.

Etapa 5: Testar a visualização de segredos

Siga as instruções em Etapa 5: Testar a visualização de segredos para exibir os segredos que você criou na etapa anterior. Use identidades SAML diferentes com atributos para corresponder às tags indicadas no tutorial ABAC.

Etapa 6: Testar a escalabilidade

Siga as instruções em Etapa 6: Testar a escalabilidade para testar a escalabilidade. Faça isso adicionando uma nova identidade ao seu IdP baseado em SAML com os seguintes atributos:

- `cost-center = 101010`
- `access-project = cen`
- `access-team = eng`

Etapa 7: Testar a atualização e a exclusão de segredos

Siga as instruções em Etapa 7: Testar a atualização e a exclusão de segredos para atualizar e excluir segredos. Use identidades SAML diferentes com atributos para corresponder às tags indicadas no tutorial ABAC.

> **Important**
>
> Exclua todos os segredos que você criou para evitar cobranças. Para obter detalhes sobre preços no Secrets Manager, consulte Preços do AWS Secrets Manager.

Resumo

Você concluiu com êxito todas as etapas necessárias para usar tags de sessão SAML e tags de recursos para o gerenciamento de permissões.

> **Note**
>
> Você adicionou políticas que permitem ações somente sob condições específicas. Se você aplicar uma política diferente para seus usuários ou funções que tenha permissões mais amplas, as ações podem não estar limitadas a exigir marcação. Por exemplo, se você conceder permissões administrativas completas a um usuário usando a política gerenciada `AdministratorAccess` da AWS, essas políticas não vão restringir esse acesso. Para obter mais informações sobre como as permissões são determinadas quando várias políticas estão envolvidas, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso.

Tutorial do IAM: Permitir que os usuários gerenciem suas credenciais e configurações de MFA

Você pode permitir que os usuários gerenciem seus próprios dispositivos e credenciais de autenticação multifator (MFA) na página Credenciais de segurança. É possível usar o Console de gerenciamento da AWS para configurar credenciais (chaves de acesso, senhas, certificados de assinatura e chaves públicas de SSH), excluir ou desativar credenciais que não sejam mais necessárias e habilitar dispositivos MFA para seus usuários. Isso é útil para um pequeno número de usuários, mas essa tarefa que pode rapidamente se tornar demorada conforme o número de usuários aumenta. Este tutorial mostra como habilitar essas melhores práticas sem sobrecarregar seus administradores.

Este tutorial mostra como permitir que usuários acessem os serviços da AWS, mas apenas quando fazem login com MFA. Se o login desse usuários não for feito com um dispositivo MFA, eles não poderão acessar outros serviços.

Esse fluxo de trabalho tem três etapas básicas.

Etapa 1: Criar uma política para impor o login com MFA

Crie uma política gerenciada pelo cliente que proíba todas as ações, exceto as poucas ações do IAM. Essas exceções permitem que um usuário altere suas próprias credenciais e gerencie seus

dispositivos de MFA na página Credenciais de segurança. Para obter mais informações sobre como acessar essa página, consulte Como os usuários do IAM alteram a própria senha (console).

Etapa 2: Anexar políticas ao grupo de usuários de teste

Crie um grupo de usuários cujos membros tenham acesso total a todas as ações do Amazon EC2 se eles fizerem login com MFA. Para criar esse grupo de usuários, anexe a política gerenciada pela AWS chamada `AmazonEC2FullAccess` e a política gerenciada pelo cliente que você criou na primeira etapa.

Etapa 3: Testar o acesso do usuário

Faça login como o usuário de teste para verificar se o acesso ao Amazon EC2 está bloqueado até que o usuário crie um dispositivo com MFA. Depois, o usuário pode fazer login usando esse dispositivo.

Pré-requisitos

Para executar as etapas neste tutorial, você já deve ter o seguinte:

- Uma Conta da AWS com a qual você possa fazer login como usuário do IAM com permissões administrativas.
- O número do ID da conta que você digitará na política na Etapa 1.

  Para localizar o número de ID de sua conta, na barra de navegação na parte superior da página, escolha Suporte e, em seguida, escolha Central de suporte. Você pode localizar o ID da conta no menu Suporte dessa página.
- Um dispositivo de MFA virtual (baseado em software), chave de segurança FIDO ou dispositivo de MFA baseado em hardware.
- Um usuário de teste do IAM que é membro de um grupo de usuários da seguinte forma:

Nome de usuário	Instruções sobre nome de usuário	Nome do grupo de usuários	Adicionar usuário como membro	Instruções sobre grupo de usuários
MFAUser	Escolha apenas a opção para Habilitar acesso	EC2MFA	MFAUser	NÃO anexe quaisquer políticas nem conceda

Nome de usuário	Instruções sobre nome de usuário	Nome do grupo de usuários	Adicionar usuário como membro	Instruções sobre grupo de usuários
	ao console: opcional e atribua uma senha.			permissões para este grupo de usuários.

Etapa 1: Criar uma política para impor o login com MFA

Comece com a criação de uma política gerenciada pelo cliente do IAM que negue todas as permissões, exceto as necessárias para os usuários do IAM gerenciarem suas próprias credenciais e dispositivos com MFA.

1. Faça login no Console de Gerenciamento da AWS como um usuário com credenciais de administrador. Para seguir as melhores práticas do IAM não faça login com as suas credenciais de Usuário raiz da conta da AWS.

   > **Important**
   >
   > As práticas recomendadas do IAM aconselham exigir que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias em vez de usuários do IAM com credenciais de longo prazo. Recomendamos usar somente usuários do IAM para casos de uso específicos não compatíveis com usuários federados.

2. Abra o console do IAM em https://console.aws.amazon.com/iam/.

3. No painel de navegação, escolha Políticas e, em seguida, Criar política.
4. Selecione a guia JSON copie o texto do documento de política JSON a seguir: AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.
5. Cole o texto da política na caixa de texto JSON. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a validação da política e depois escolha Avançar.

> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. No entanto, a política acima inclui o elemento `NotAction`, que não é compatível com o editor visual. Para esta política, você verá uma notificação na guia Editor visual. Volte para a guia JSON para continuar a trabalhar com essa política.
> Este exemplo de polítia não permite que os usuários redefinam uma senha quando fazem login no Console de gerenciamento da AWS pela primeira vez. Recomendamos que você não conceda permissões a novos usuários até que eles façam login e redefinam suas senhas.

6. Na página Revisar e criar, digite **`Force_MFA`** para o nome da política. Para a descrição da política, digite **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`** Na área Tags, você tem a opção de adicionar pares de chave-valor de tag à política gerenciada pelo cliente. Revise as permissões concedidas pela política e depois escolha Criar política para salvar seu trabalho.

   A nova política aparece na lista de políticas gerenciadas e está pronta para ser anexada.

Etapa 2: Anexar políticas ao grupo de usuários de teste

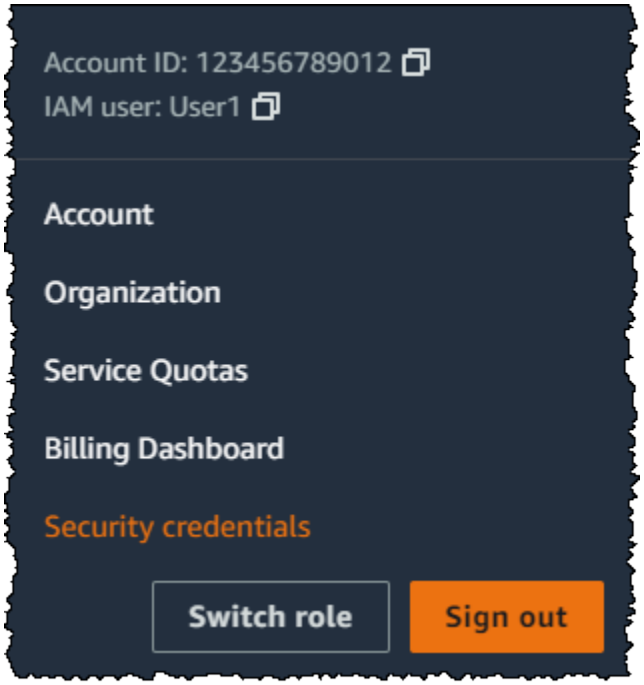
Em seguida, anexe duas políticas ao grupo de usuários de teste do IAM, que serão usadas para conceder as permissões protegidas por MFA.

1. No painel de navegação, selecione User groups (Grupos de usuários).
2. Na caixa de pesquisa, digite **`EC2MFA`** e, em seguida, escolha o nome do grupo (não a caixa de seleção) na lista.
3. Escolha a guia Permissões, escolha Adicionar permissões e depois Anexar políticas.
4. Na página Attach permission policies to EC2MFA group (Anexar políticas de permissão ao grupo EC2MFA), na caixa de pesquisa, digite **`EC2Full`**. Depois, marque a caixa de seleção ao lado de AmazonEC2FullAccess na lista. Não salve ainda as alterações.
5. Na caixa de pesquisa, digite **`Force`** e, em seguida, marque a caixa de seleção ao lado de Force_MFA na lista.
6. Escolha Anexar políticas.

Etapa 3: Testar o acesso do usuário

Nesta parte do tutorial, você faz login como o usuário de teste e verifica se a política funciona conforme o esperado.

1. Faça login na sua Conta da AWS como **`MFAUser`** com a senha atribuída na seção anterior. Use o URL: `https://<alias or account ID number>.signin.aws.amazon.com/console`
2. Escolha EC2 para abrir o console do Amazon EC2 e confirme se o usuário não tem permissões para fazer nada.
3. Na barra de navegação no canto superior direito, selecione o nome de usuário `MFAUser` e selecione Security Credentials (Credenciais de segurança).



4. Agora adicione um dispositivo MFA. Na seção Multi-Factor Authentication (MFA), selecione Assign MFA device (Atribuir dispositivo MFA).

> **Note**
>
> É possível que você receba um erro informando que você não está autorizado a executar `iam:DeleteVirtualMFADevice`. Isso pode acontecer se alguém tiver começado a atribuir um dispositivo MFA virtual a esse usuário anteriormente e tiver cancelado o processo. Para continuar, você ou outro administrador deve excluir

o dispositivo de MFA virtual existente e não atribuído do usuário. Para obter mais informações, consulte Não estou autorizado a executar: iam:DeleteVirtualMFADevice.

5. Para este tutorial, usamos um dispositivo MFA virtual (em software), como o aplicativo Google Authenticator em um celular. Escolha a Authenticator app (Aplicação de autenticador) e clique em Next (Avançar).

   O IAM gera e exibe informações de configuração para o dispositivo com MFA virtual, incluindo um código QR gráfico. O gráfico é uma representação da chave de configuração secreta que está disponível para entrada manual em dispositivos que não suportam códigos de QR.

6. Abra o seu aplicativo de MFA virtual. (Para obter uma lista de aplicativos que você pode usar para hospedar dispositivos MFA virtuais, consulte Aplicativos de MFA virtual.) Se o aplicativo de MFA virtual oferecer suporte a várias contas (vários dispositivos MFA virtuais), selecione a opção para criar uma nova conta (um novo dispositivo MFA virtual).

7. Determine se o aplicativo de MFA é compatível com códigos QR e, em seguida, execute uma das seguintes ações:

   - No assistente, escolha Mostrar código QR. Em seguida, use o aplicativo para digitalizar o código QR. Por exemplo, você pode escolher o ícone de câmera ou escolher uma opção semelhante a Digitalizar código e, em seguida, usar a câmera do dispositivo para digitalizar o código.
   - No assistente Set up device (Configurar dispositivo), selecione Show secret key (Exibir chave secreta) e digite a chave secreta em sua aplicação de MFA.

   Quando você tiver concluído, o dispositivo MFA virtual inicia a geração de senhas de uso único.

8. No assistente Set up device (Configurar dispositivo), na caixa Enter the code from your authenticator app. (Insira o código da aplicação de autenticador.), digite a senha de uso único que atualmente é exibida no dispositivo de MFA virtual. Escolha Register MFA (Registrar MFA).

   Important

   Envie sua solicitação imediatamente após gerar o código. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA conseguirá se associar ao usuário. No entanto, o dispositivo MFA estará fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Caso isso ocorra, você pode ressincronizar o dispositivo.

O dispositivo MFA virtual está pronto para ser usado com a AWS.

9. Saia do console e, em seguida, faça login como **`MFAUser`** novamente. Dessa vez, a AWS solicita um código de MFA de seu telefone. Quando você obtiver esse código, digite-o na caixa e, em seguida, escolha Enviar.
10. Escolha EC2 para abrir o console do Amazon EC2 novamente. Desta vez, observe que você poderá ver todas as informações e realizar qualquer ação desejada. Se você acessar qualquer outro console como esse usuário, verá mensagens de acesso negado. O motivo é que as políticas neste tutorial concedem acesso somente ao Amazon EC2.

Recursos relacionados

Para obter informações adicionais, consulte os seguintes tópicos:

- AWSCódigo da autenticação multifator no IAM da
- Login habilitado para MFA

Tutorial do IAM: use um modelo do CloudFormation para criar um provedor de identidades (IdP) do SAML

Para configurar a federação SAML para sua conta da AWS, você precisa criar um provedor de identidades (IdP) SAML. Este tutorial mostra como usar um modelo do CloudFormation para criar um IdP do SAML que estabeleça confiança entre a AWS e seu IdP externo.

O modelo cria um IdP do SAML configurado com o documento de metadados do seu IdP. Os perfis federados do IAM podem então fazer referência a esse IdP para permitir que usuários autenticados do seu IdP externo acessem os recursos da AWS.

O recurso implantado consiste em um IdP do SAML configurado com o documento de metadados do seu IdP e configurações de criptografia opcionais.

Pré-requisitos

Este tutorial pressupõe que você já tenha os seguintes itens configurados:

- Python 3.6 ou posterior instalado em sua máquina local para executar o comando Python usado neste tutorial para formatar o arquivo XML de metadados SAML do seu IdP.

- Um documento de metadados SAML do seu IdP externo salvo como um arquivo XML.

Criar um IdP do SAML usando o CloudFormation

Para criar o IdP do SAML, crie um modelo do CloudFormation e use-o para criar uma pilha contendo o recurso IdP.

Criar o modelo do

Primeiro, crie o modelo do CloudFormation.

1. Na seção Modelo, escolha o ícone de cópia na guia JSON ou YAML para copiar o conteúdo do modelo.
2. Copie o conteúdo do modelo em um novo arquivo.
3. Salve o arquivo localmente.

Crie a pilha.

Em seguida, use o arquivo de modelo que você salvou para provisionar uma pilha do CloudFormation.

1. Abra o console do CloudFormation em https://console.aws.amazon.com/cloudformation.
2. Na página Pilhas, no menu Criar pilha, escolha com novos recursos (padrão).
3. Especifique o modelo:
    a. Em Pré-requisito, escolha Escolher um modelo existente.
    b. Em Especificar modelo, escolha Fazer upload de um arquivo de modelo.
    c. Selecione Escolher arquivo, navegue até o arquivo de modelo e selecione-o.
    d. Escolha Próximo.
4. Especifique os seguintes detalhes da pilha:
    a. Digite um nome de pilha.
    b. Você pode deixar o campo IdentityProviderName em branco para gerar automaticamente um nome com base no nome da pilha ou inserir um nome personalizado para seu IdP do SAML. Nomes personalizados devem conter apenas caracteres alfanuméricos, pontos, sublinhados e hifens.

c. Para IdentityProviderSAMLMetadataDocument, você precisa formatar seu arquivo XML de metadados SAML como uma única linha antes de colá-lo nesse campo. Isso é necessário porque o console do CloudFormation exige que o conteúdo XML seja formatado como uma única linha quando passado pelos parâmetros do console.

Use o seguinte comando Python para reformatar seu arquivo XML:

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>
\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-
metadata.xml
```

> **Note**
>
> O documento de metadados SAML do IdP deve ser formatado como uma única linha para entrada de parâmetros do console. O comando Python remove quebras de linha e espaços em branco extras para criar o formato necessário, mantendo todo o conteúdo e estrutura originais.

Copie a saída do comando Python e cole-a no campo IdentityProviderSAMLMetadataDocument.

Exemplo de documento de metadados SAML formatado (abreviado):

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
```

```
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. Para outros parâmetros, aceite os valores padrão ou insira seus próprios com base em seus requisitos:

- IdentityProviderAddPrivateKey: chave privada opcional para descriptografar asserções do SAML
- IdentityProviderAssertionEncryptionMode: opcional, define o modo de criptografia para asserções SAML (permitido, obrigatório ou vazio)

e. Escolha Próximo.

5. Configurar as opções da pilha:

a. Em Opções de falha de pilha, escolha Excluir todos os recursos recém-criados.

> **Note**
>
> A escolha dessa opção evita que você seja cobrado por recursos cuja política de exclusão especifica que eles sejam retidos mesmo que a criação da pilha falhe.

b. Aceite todos os outros valores padrão.

c. Em Capacidades, marque a caixa para confirmar que o CloudFormation pode criar recursos do IAM em sua conta.

d. Escolha Próximo.

6. Revise os detalhes a pilha e selecione Enviar.

CloudFormationO cria a pilha. Depois que a criação da pilha for concluída, os recursos da pilha estarão prontos para uso. Você pode usar a guia Recursos na página de detalhes da pilha para visualizar os recursos que foram provisionados em sua conta.

A pilha produzirá os seguintes valores, que você pode ver na guia Saídas:

- ProviderARN: o ARN do IdP do SAML criado (por exemplo `arn:aws:iam::123456789012:saml-provider/CompanyIdP`). Você precisará desse ARN ao criar perfis que confiem nesse provedor.
- ProviderName: o nome do IdP do SAML criado (por exemplo `CompanyIdP`, se você especificou um nome personalizado ou `my-saml-stack-saml-provider` se usou a nomenclatura padrão).

Essas saídas também são exportadas, permitindo que sejam importadas por outras pilhas do CloudFormation usando o função `Fn::ImportValue`.

Verificar o IdP do SAML

Assim que o IdP do SAML for criado, você poderá verificar sua configuração e anotar seu ARN para uso com perfis federados.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Identity providers (Provedores de identidade).

   Você deverá ver seu IdP do SAML recém-criado na lista.
3. Selecione o nome do IdP para visualizar seus detalhes.

   Na página de detalhes do IdP, você pode ver o documento de metadados do SAML e outros detalhes de configuração.
4. Observe o ARN do provedor exibido na página de detalhes.

   Você precisará desse ARN ao criar perfis federados do IAM que confiam nesse IdP.
5. Revise o documento de metadados para certificar-se de que corresponda ao que você forneceu do IdP externo.

Seu IdP do SAML agora está pronto para ser usado por perfis federadas do IAM. Você pode criar perfis que confiem nesse IdP para permitir que usuários autenticados do seu IdP externo assumam esses perfis e acessem os recursos da AWS.

Limpeza: excluir recursos

Como etapa final, você excluirá a pilha e os recursos que ela contém.

1. Abra o CloudFormation Console.
2. Na página Pilhas, escolha a pilha criada a partir do modelo, escolha Excluir e confirme para Excluir.

   O CloudFormation inicia a exclusão da pilha e de todos os recursos que ela inclui.

Detalhes do modelo do CloudFormation

Recursos

O modelo do CloudFormation para este tutorial criará o seguinte recurso em sua conta:

`AWS::IAM::SAMLProvider`: um IdP do SAML que estabeleça confiança entre AWS e seu IdP externo.

Configuração

O modelo inclui os seguintes parâmetros configuráveis:

- IdentityProviderName: o nome do seu IdP do SAML (deixe em branco para nome gerado automaticamente)

  Exemplo: `CompanyIdP` ou `EnterpriseSSO`
- IdentityProviderSAMLMetadataDocument: o documento de metadados SAML do seu IdP externo (formatado como uma única linha)
- IdentityProviderAddPrivateKey: chave privada opcional para descriptografar asserções do SAML
- IdentityProviderAssertionEncryptionMode: opcional, define o modo de criptografia para asserções SAML

Modelo do CloudFormation

Salve o código JSON ou YAML a seguir em outro arquivo para usar como modelo do CloudFormation neste tutorial.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-idp",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
```

```
      "ConstraintDescription": "Must be empty or contain only alphanumeric
characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    }
  },
  "Conditions": {
    "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
    "HasEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]},
    "HasCustomName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "IdentityProviderName"},
""]}]}
  },
  "Resources": {
    "SAMLProvider": {
      "Type": "AWS::IAM::SAMLProvider",
      "Properties": {
        "Name": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Ref": "AWS::NoValue"}]},
        "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
{"Fn::Sub": "${AWS::StackName}-saml-provider"}]}
          }
        ],
```

```
        "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
 "IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
        "AssertionEncryptionMode": {"Fn::If": ["HasEncryptionMode", {"Ref":
 "IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
      }
    }
  },
  "Outputs": {
    "ProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderARN"}
      }
    },
    "ProviderName": {
      "Description": "Name of the SAML Identity Provider",
      "Value": {"Fn::If": ["HasCustomName", {"Ref": "IdentityProviderName"},
 {"Fn::Sub": "${AWS::StackName}-saml-provider"}]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-ProviderName"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens'

  IdentityProviderSAMLMetadataDocument:
    Type: String
```

```
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
 key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
 decrypt SAML assertions.
    Default: ""

  IdentityProviderAssertionEncryptionMode:
    Type: String
    Description: Optional, sets encryption mode for SAML assertions
    Default: ""
    AllowedValues:
      - ""
      - "Allowed"
      - "Required"

Conditions:
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasEncryptionMode: !Not [!Equals [!Ref IdentityProviderAssertionEncryptionMode,
 ""]]
  HasCustomName: !Not [!Equals [!Ref IdentityProviderName, ""]]

Resources:
  SAMLProvider:
    Type: 'AWS::IAM::SAMLProvider'
    Properties:
      Name: !If
        - HasCustomName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      Tags:
        - Key: Name
          Value: !If
            - HasCustomName
            - !Ref IdentityProviderName
            - !Sub '${AWS::StackName}-saml-provider'
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
```

```
        - HasEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
        - !Ref AWS::NoValue

Outputs:
  ProviderARN:
    Description: 'ARN of the created SAML Identity Provider'
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-ProviderARN'

  ProviderName:
    Description: 'Name of the SAML Identity Provider'
    Value: !If
      - HasCustomName
      - !Ref IdentityProviderName
      - !Sub '${AWS::StackName}-saml-provider'
    Export:
      Name: !Sub '${AWS::StackName}-ProviderName'
```

Tutorial do IAM: use um modelo do CloudFormation para criar um perfil do IAM federado por SAML

Quando você tem um provedor de identidades (IdP do SAML) existente configurado em sua conta da AWS, você pode criar perfis federados do IAM que confiam nesse IdP. Este tutorial mostra como usar um modelo do CloudFormation para criar um perfil do IAM federado por SAML que pode ser assumido por usuários autenticados por meio do seu IdP externo.

O modelo cria um perfil federado do IAM com uma política de confiança que permite que o IdP do SAML assuma o perfil. Os usuários autenticados pelo seu IdP externo podem assumir esse perfil para acessar recursos da AWS nas permissões anexadas ao perfil.

O recurso implantado consiste no seguinte:

- Um perfio federada do IAM que confia em seu IdP do SAML existente.
- Políticas gerenciadas configuráveis que podem ser anexadas ao perfil para conceder permissões específicas.
- Configurações opcionais de limite de permissões e duração da sessão.

Pré-requisitos

Este tutorial pressupõe que você já tenha os seguintes itens configurados:

- Um IdP do SAML existente configurado em sua conta da AWS. Se você não tiver um, poderá criá-lo usando o tutorial Tutorial do IAM: use um modelo do CloudFormation para criar um provedor de identidades (IdP) do SAML.
- O ARN do seu IdP do SAML, que você precisará especificar como parâmetro ao criar a pilha.
- Python 3.6 ou posterior instalado em sua máquina local para executar o comando Python usado neste tutorial para formatar o arquivo XML de metadados SAML do seu IdP.

Crie um perfil federado do SAML usando CloudFormation

Para criar o perfil federado por SAML, você criará um modelo do CloudFormation e o usará para criar uma pilha contendo o perfil.

Criar o modelo do

Primeiro, crie o modelo do CloudFormation.

1. Na seção Modelo, clique no ícone de cópia na guia JSON ou YAML para copiar o conteúdo do modelo.
2. Copie o conteúdo do modelo em um novo arquivo.
3. Salve o arquivo localmente.

Crie a pilha.

Em seguida, use o arquivo de modelo que você salvou para provisionar uma pilha do CloudFormation.

1. Abra o console do CloudFormation em https://console.aws.amazon.com/cloudformation.
2. Na página Pilhas, no menu Criar pilha, escolha com novos recursos (padrão).
3. Especifique o modelo:
    a. Em Pré-requisito, escolha Escolher um modelo existente.
    b. Em Especificar modelo, escolha Fazer upload de um arquivo de modelo.
    c. Selecione Escolher arquivo, navegue até o arquivo de modelo e selecione-o.

d. Escolha Próximo.

4. Especifique os seguintes detalhes da pilha:

   a. Digite um nome de pilha.

   b. Para SAMLProviderARN, insira o ARN do seu IdP do SAML existente. Isso deve estar no formato `arn:aws:iam::123456789012:saml-provider/YourProviderName`.

      Exemplo: `arn:aws:iam::123456789012:saml-provider/CompanyIdP`

      > **Note**
      >
      > Se você criou seu IdP do SAML usando o tutorial Tutorial do IAM: use um modelo do CloudFormation para criar um provedor de identidades (IdP) do SAML, poderá encontrar o ARN do provedor na guia Saídas dessa pilha do CloudFormation.

   c. Para RoleName, é possível deixar isso em branco para gerar automaticamente um nome com base no nome da pilha ou inserir um nome personalizado para o perfil do IAM.

      Exemplo: `SAML-Developer-Access` ou `SAML-ReadOnly-Role`

   d. Para outros parâmetros, aceite os valores padrão ou insira seus próprios com base em seus requisitos:

      - RoleSessionDuration: duração máxima da sessão em segundos (3600-43200, padrão 7200)

        Exemplo: `14400` (4 horas)

      - RolePermissionsBoundary: ARN opcional de uma política de limite de permissões

        Exemplo: `arn:aws:iam::123456789012:policy/DeveloperBoundary`

      - RolePath: caminho para o perfil do IAM (o padrão é /)

        Exemplo: `/saml-roles/`

      - ManagedPolicy1-5: ARNs opcionais de até 5 políticas gerenciadas para anexar

        Exemplo de ManagedPolicy1: `arn:aws:iam::aws:policy/ReadOnlyAccess`

        Exemplo de ManagedPolicy2: `arn:aws:iam::123456789012:policy/CustomPolicy`

e. Escolha Próximo.

5. Configurar as opções da pilha:

   a. Em Opções de falha de pilha, escolha Excluir todos os recursos recém-criados.

   > **Note**
   >
   > A escolha dessa opção evita que você seja cobrado por recursos cuja política de exclusão especifica que eles sejam retidos mesmo que a criação da pilha falhe.

   b. Aceite todos os outros valores padrão.

   c. Em Capacidades, marque a caixa para confirmar que o CloudFormation pode criar recursos do IAM em sua conta.

   d. Escolha Próximo.

6. Revise os detalhes a pilha e selecione Enviar.

CloudFormationO cria a pilha. Quando a criação da pilha for concluída, os recursos da pilha estarão prontos para uso. Você pode usar a guia Recursos na página de detalhes da pilha para visualizar os recursos que foram provisionados em sua conta.

A pilha produzirá o seguinte valor, que você pode ver na guia Saídas:

- RoleARN: o ARN do perfil do IAM criado (por exemplo, `arn:aws:iam::123456789012:role/SAML-Developer-Access` ou `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4` se estiver usando um nome gerado automaticamente).

Você precisará desse ARN de perfil ao configurar seu IdP para enviar os atributos SAML apropriados para assumir o perfil.

Teste o perfil federado do SAML

Depois que o perfil federado do SAML for criado, você pode verificar sua configuração e testar a configuração da federação.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Encontre e selecione o perfil federado recém-criado.

Se você forneceu um nome de perfil personalizado, procure esse nome. Se você deixar o parâmetro RoleName vazio, o perfil terá um nome gerado automaticamente com base no nome da pilha e um identificador exclusivo.

4. Selecione a guia Relações de confiança para revisar a política de confiança.

   A política de confiança deve mostrar que o IdP do SAML é confiável para assumir esse perfil com a condição de que o público do SAML (`SAML:aud`) corresponda ao `https://signin.aws.amazon.com/saml`.

5. Clique na guia Permissões para revisar as políticas anexadas.

   Você pode ver todas as políticas gerenciadas que foram anexadas ao perfil durante a criação.

6. Observe o ARN do perfil exibido na página de resumo do perfil.

   Você precisará desse ARN para configurar seu IdP externo para permitir que os usuários assumam esse perfil.

Seu perfil federado do SAML agora está pronto para ser usado. Configure seu IdP externo para incluir o ARN desse perfil nas asserções do SAML, e os usuários autenticados poderão assumir esse perfil para acessar recursos da AWS.

Limpeza: excluir recursos

Como etapa final, você excluirá a pilha e os recursos que ela contém.

1. Abra o CloudFormation Console.
2. Na página Pilhas, escolha a pilha criada a partir do modelo, escolha Excluir e confirme para Excluir.

   O CloudFormation inicia a exclusão da pilha e de todos os recursos que ela inclui.

Detalhes do modelo do CloudFormation

Recursos

O modelo do CloudFormation para este tutorial criará o seguinte recurso em sua conta:

- `AWS::IAM::Role`: uma perfil federado do IAM que pode ser assumido por usuários autenticados por meio do seu IdP do SAML.

Configuração

O modelo inclui os seguintes parâmetros configuráveis:

- RoleName: nome do perfil do IAM (deixe em branco para nome gerado automaticamente)
- SAMLProviderARN: ARN do IdP do SAML (obrigatório)
- RoleSessionDuration: duração máxima da sessão em segundos (3600-43200, padrão 7200)
- RolePermissionsBoundary: ARN opcional da política de limite de permissões
- RolePath: caminho para o perfil do IAM (padrão /)
- ManagedPolicy1-5: ARNs opcionais de até 5 políticas gerenciadas para anexar

Modelo do CloudFormation

Salve o código JSON ou YAML a seguir em outro arquivo para usar como modelo do CloudFormation neste tutorial.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "[AWSDocs] IAM: tutorial_saml-federated-role",
  "Parameters": {
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
 '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-"
    },
    "SAMLProviderARN": {
      "Type": "String",
      "Description": "ARN of the SAML Identity Provider",
      "AllowedPattern": "^arn:aws:iam::\\d{12}:saml-provider/[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be a valid SAML provider ARN"
```

```
    },
    "RoleSessionDuration": {
      "Type": "Number",
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RolePermissionsBoundary": {
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM role (must start and end with /)",
      "Default": "/",
      "AllowedPattern": "^\/.*\/$|^\/$",
      "ConstraintDescription": "Role path must start and end with forward slash (/)"
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
```

```
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
    "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
    "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
    "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
    "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
    "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
    "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"}, ""]}]}
  },
  "Resources": {
    "SAMLFederatedRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
        "Description": "IAM role with SAML provider trust",
        "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
        "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
        "Path": {"Ref": "RolePath"},
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProviderARN"}
              },
              "Action": "sts:AssumeRoleWithSAML",
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
```

```
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
 "AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
 "AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-federated-role'
```

```
Parameters:
  RoleName:
    Type: String
    Description: 'Name of the IAM Role (leave empty for auto-generated name like
 ''{StackName}-{UniqueId}'')'
    Default: ""
    AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
    ConstraintDescription: 'Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-'

  SAMLProviderARN:
    Type: String
    Description: 'ARN of the SAML Identity Provider'
    AllowedPattern: '^arn:aws:iam::\d{12}:saml-provider/[a-zA-Z0-9._-]+$'
    ConstraintDescription: 'Must be a valid SAML provider ARN'

  RoleSessionDuration:
    Type: Number
    Description: 'The maximum session duration (in seconds) that you want to set for
 the specified role (3600-43200)'
    MinValue: 3600
    MaxValue: 43200
    Default: 7200

  RolePermissionsBoundary:
    Type: String
    Description: Optional ARN of the permissions boundary policy (leave empty for
 none)
    Default: ""

  RolePath:
    Type: String
    Description: 'Path for the IAM role (must start and end with /)'
    Default: "/"
    AllowedPattern: '^\/.*\/$|^\/$'
    ConstraintDescription: 'Role path must start and end with forward slash (/)'

  RoleManagedPolicy1:
    Type: String
    Description: Optional managed policy ARN 1
    Default: ""
  RoleManagedPolicy2:
    Type: String
    Description: Optional managed policy ARN 2
```

```
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]

Resources:
  SAMLFederatedRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Description: 'IAM role with SAML provider trust'
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProviderARN
```

```
            Action: 'sts:AssumeRoleWithSAML'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns:
        !Split
          - ','
          - !Join
            - ','
            - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref 'AWS::NoValue']
              - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref 'AWS::NoValue']
              - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref 'AWS::NoValue']
              - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref 'AWS::NoValue']
              - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref 'AWS::NoValue']

Outputs:
  RoleARN:
    Description: 'ARN of the created IAM Role'
    Value: !GetAtt SAMLFederatedRole.Arn
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'
```

Tutorial do IAM: use um modelo do CloudFormation para criar um provedor de identidades (IdP) do SAML e um perfil do IAM federado por SAML

Para se familiarizar com a federação por SAML e seus recursos, você usará um modelo do CloudFormation para configurar um provedor de identidades (IdP) do SAML e um perfil do IAM federado associado. Este tutorial mostra como criar os dois recursos juntos em uma única pilha.

O modelo cria um IdP do SAML que pode ser usado para acesso federado aos recursos da AWS, junto com um perfil do IAM que confia no provedor de SAML. Os usuários autenticados pelo seu IdP externo podem assumir esse perfil para acessar recursos da AWS.

Os recursos implantados consistem no seguinte:

- Um IdP do SAML configurado com o documento de metadados do seu IdP.
- Um perfil federado do IAM que confia no IdP do SAML e pode ser assumido por usuários autenticados.

- Políticas gerenciadas configuráveis que podem ser anexadas ao perfil para conceder permissões específicas.

Pré-requisitos

Este tutorial pressupõe que você já tenha os seguintes itens configurados:

- Python 3.6 ou posterior instalado em sua máquina local para executar o comando Python usado neste tutorial para formatar o arquivo XML de metadados SAML do seu IdP.
- Um documento de metadados SAML do seu IdP externo salvo como um arquivo XML.

Criar um IdP e um perfil do SAML usando o CloudFormation

Para criar o perfil federado e o IdP do SAML, você criará um modelo do CloudFormation e o usará para criar uma pilha contendo os dois recursos.

Criar o modelo do

Primeiro, crie o modelo do CloudFormation.

1. Na seção Modelo, clique no ícone de cópia na guia JSON ou YAML para copiar o conteúdo do modelo.
2. Copie o conteúdo do modelo em um novo arquivo.
3. Salve o arquivo localmente.

Crie a pilha.

Em seguida, use o arquivo de modelo que você salvou para provisionar uma pilha do CloudFormation.

1. Abra o console do CloudFormation em https://console.aws.amazon.com/cloudformation.
2. Na página Pilhas, no menu Criar pilha, escolha com novos recursos (padrão).
3. Especifique o modelo:
    a. Em Pré-requisito, escolha Escolher um modelo existente.
    b. Em Especificar modelo, escolha Fazer upload de um arquivo de modelo.

c. Selecione Escolher arquivo, navegue até o arquivo de modelo e selecione-o.

d. Escolha Próximo.

4. Especifique os seguintes detalhes da pilha:

a. Digite um nome de pilha.

b. Você pode deixar o campo IdentityProviderName em branco para gerar automaticamente um nome com base no nome da pilha ou inserir um nome personalizado para seu IdP do SAML.

Exemplo: `CompanyIdP` ou `EnterpriseSSO`

c. Para IdentityProviderSAMLMetadataDocument, você precisa formatar seu arquivo XML de metadados SAML como uma única linha antes de colá-lo nesse campo. Isso é necessário porque o console do CloudFormation exige que o conteúdo XML seja formatado como uma única linha quando passado pelos parâmetros do console.

Use o seguinte comando Python para reformatar seu arquivo XML:

```
python3 -c "import sys, re; content=open(sys.argv[1]).read(); print(re.sub(r'>
\s+<', '><', content.replace('\n', '').replace('\r', '').strip()))" saml-
metadata.xml
```

> **Note**
>
> O documento de metadados SAML do IdP deve ser formatado como uma única linha para entrada de parâmetros do console. O comando Python remove quebras de linha e espaços em branco extras para criar o formato necessário, mantendo todo o conteúdo e estrutura originais.

Copie a saída do comando Python e cole-a no campo IdentityProviderSAMLMetadataDocument.

Exemplo de documento de metadados SAML formatado (abreviado):

```
<?xml version="1.0" encoding="UTF-8"?><md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 entityID="https://portal.sso.example.com/saml/assertion/
CompanyIdP"><md:IDPSSODescriptor WantAuthnRequestsSigned="false"
```

```
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor
 use="signing"><ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/
xmldsig#"><ds:X509Data><ds:X509Certificate>MIIDXTCCAkWgAwIBAgIJAJC1HiIAZAiIMA0GCSqGSIb3
ds:X509Certificate></ds:X509Data></ds:KeyInfo></
md:KeyDescriptor><md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="https://portal.sso.example.com/saml/logout/
CompanyIdP"/><md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat><md:SingleSignOnService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://
portal.sso.example.com/saml/assertion/CompanyIdP"/></md:IDPSSODescriptor></
md:EntityDescriptor>
```

d. Você pode deixar RoleName em branco para gerar automaticamente um nome com base no nome da pilha ou inserir um nome personalizado para o perfil federado do IAM.

   Exemplo: `SAML-Developer-Access` ou `SAML-ReadOnly-Role`

e. Para outros parâmetros, aceite os valores padrão ou insira seus próprios com base em seus requisitos:

   - IdentityProviderAddPrivateKey: chave privada opcional para descriptografar asserções do SAML
   - IdentityProviderAssertionEncryptionMode: modo de criptografia para asserções SAML

     Valores de exemplo: `Allowed`, `Required` ou deixe em branco para sem criptografia
   - RoleSessionDuration: duração máxima da sessão em segundos (3600-43200, padrão 7200)

     Exemplo: `14400` (4 horas)
   - RolePermissionsBoundary: ARN opcional de uma política de limite de permissões

     Exemplo: `arn:aws:iam::123456789012:policy/DeveloperBoundary`
   - RolePath: caminho para o perfil do IAM (o padrão é /)

     Exemplo: `/saml-roles/`
   - RoleManagedPolicy1-5: ARNs opcionais de até 5 políticas gerenciadas para anexar

     Exemplo de RoleManagedPolicy1: `arn:aws:iam::aws:policy/ReadOnlyAccess`

Exemplo de RoleManagedPolicy2: `arn:aws:iam::123456789012:policy/CustomPolicy`

f. Escolha Próximo.

5. Configurar as opções da pilha:

a. Em Opções de falha de pilha, escolha Excluir todos os recursos recém-criados.

> **Note**
>
> A escolha dessa opção evita que você seja cobrado por recursos cuja política de exclusão especifica que eles sejam retidos mesmo que a criação da pilha falhe.

b. Aceite todos os outros valores padrão.

c. Em Capacidades, marque a caixa para confirmar que o CloudFormation pode criar recursos do IAM em sua conta.

d. Escolha Próximo.

6. Revise os detalhes a pilha e selecione Enviar.

CloudFormationO cria a pilha. Quando a criação da pilha for concluída, os recursos da pilha estarão prontos para uso. Você pode usar a guia Recursos na página de detalhes da pilha para visualizar os recursos que foram provisionados em sua conta.

A pilha produzirá os seguintes valores, que você pode ver na guia Saídas:

- RoleARN: o ARN do perfil do IAM criado (por exemplo, `arn:aws:iam::123456789012:role/SAML-Developer-Access` ou `arn:aws:iam::123456789012:role/stack-name-a1b2c3d4` se estiver usando um nome gerado automaticamente).
- IdentityProviderARN: o ARN do IdP do SAML criado (por exemplo, `arn:aws:iam::123456789012:saml-provider/CompanyIdP`).

Você precisará desses dois ARNs ao configurar seu IdP para enviar os atributos de SAML apropriados para assumir o perfil.

Testar a federação do SAML

Depois que o IdP do SAML e o perfil federado forem criados, você poderá testar a configuração da federação.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Identity providers (Provedores de identidade).

   Você deverá ver seu IdP do SAML recém-criado na lista.
3. Selecione o nome do IdP para visualizar seus detalhes.

   Na página de detalhes do IdP, você pode ver o documento de metadados do SAML e outros detalhes de configuração.
4. No painel de navegação, escolha Perfis.
5. Encontre e selecione seu perfil federado recém-criado.

   Na página de detalhes do perfil, você pode ver a política de confiança que permite que o IdP do SAML assuma esse perfil.
6. Selecione a guia Relações de confiança para revisar a política de confiança.

   A política de confiança deve mostrar que o IdP do SAML é confiável para assumir esse perfil com a condição de que o público do SAML (`SAML:aud`) corresponda ao `https://signin.aws.amazon.com/saml`.

Limpeza: excluir recursos

Como etapa final, você excluirá a pilha e os recursos que ela contém.

1. Abra o CloudFormation Console.
2. Na página Pilhas, escolha a pilha criada a partir do modelo, escolha Excluir e confirme para Excluir.

   O CloudFormation inicia a exclusão da pilha e de todos os recursos que ela inclui.

Detalhes do modelo do CloudFormation

Recursos

O modelo do CloudFormation para este tutorial criará os seguintes recursos em sua conta:

- `AWS::IAM::SAMLProvider`: um IdP do SAML que estabeleça confiança entre AWS e seu IdP externo.
- `AWS::IAM::Role`: um perfil federado do IAM que pode ser assumido por usuários autenticados por meio do IdP do SAML.

Configuração

O modelo inclui os seguintes parâmetros configuráveis:

- IdentityProviderName: nome do IdP do SAML (deixe em branco para nome gerado automaticamente)
- IdentityProviderSAMLMetadataDocument: Documento de metadados SAML do seu IdP (obrigatório)
- IdentityProviderAddPrivateKey: chave privada opcional para descriptografar asserções do SAML
- IdentityProviderAssertionEncryptionMode: modo de criptografia para asserções SAML
- RoleName: nome do perfil do IAM (deixe em branco para nome gerado automaticamente)
- RolePath: caminho para o perfil do IAM (padrão /)
- RolePermissionsBoundary: ARN opcional da política de limite de permissões
- RoleSessionDuration: duração máxima da sessão em segundos (3600-43200, padrão 7200)
- RoleManagedPolicy1-5: ARNs opcionais de até 5 políticas gerenciadas para anexar

Modelo do CloudFormation

Salve o código JSON ou YAML a seguir em outro arquivo para usar como modelo do CloudFormation neste tutorial.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
```

```
  "Description": "[AWSDocs] IAM: tutorial_saml-idp-and-federated-role",
  "Parameters": {
    "IdentityProviderName": {
      "Type": "String",
      "Description": "Name of the SAML Identity Provider (leave empty for auto-
generated name like '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[a-zA-Z0-9._-]+$",
      "ConstraintDescription": "Must be empty or contain only alphanumeric
 characters, periods, underscores, and hyphens"
    },
    "IdentityProviderSAMLMetadataDocument": {
      "Type": "String",
      "Description": "SAML metadata document from identity provider"
    },
    "IdentityProviderAddPrivateKey": {
      "Type": "String",
      "Description": "Optional private key for decrypting SAML assertions. The
 private key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm
 to decrypt SAML assertions.",
      "Default": ""
    },
    "IdentityProviderAssertionEncryptionMode": {
      "Type": "String",
      "Description": "Optional, sets encryption mode for SAML assertions",
      "Default": "",
      "AllowedValues": ["", "Allowed", "Required"]
    },
    "RoleName": {
      "Type": "String",
      "Description": "Name of the IAM Role (leave empty for auto-generated name like
 '{StackName}-{UniqueId}')",
      "Default": "",
      "AllowedPattern": "^$|^[\\w+=,.@-]{1,64}$",
      "ConstraintDescription": "Must be empty or 1-64 characters and can contain
 alphanumeric characters and +=,.@-"
    },
    "RolePath": {
      "Type": "String",
      "Description": "Path for the IAM Role",
      "AllowedPattern": "(^\\/$)|(^\\/.*\\/$)",
      "Default": "/"
    },
    "RolePermissionsBoundary": {
```

```
      "Type": "String",
      "Description": "Optional ARN of the permissions boundary policy (leave empty
for none)",
      "Default": ""
    },
    "RoleSessionDuration": {
      "Description": "The maximum session duration (in seconds) that you want to set
for the specified role (3600-43200)",
      "Type": "Number",
      "MinValue": 3600,
      "MaxValue": 43200,
      "Default": 7200
    },
    "RoleManagedPolicy1": {
      "Type": "String",
      "Description": "Optional managed policy ARN 1",
      "Default": ""
    },
    "RoleManagedPolicy2": {
      "Type": "String",
      "Description": "Optional managed policy ARN 2",
      "Default": ""
    },
    "RoleManagedPolicy3": {
      "Type": "String",
      "Description": "Optional managed policy ARN 3",
      "Default": ""
    },
    "RoleManagedPolicy4": {
      "Type": "String",
      "Description": "Optional managed policy ARN 4",
      "Default": ""
    },
    "RoleManagedPolicy5": {
      "Type": "String",
      "Description": "Optional managed policy ARN 5",
      "Default": ""
    }
  },
  "Conditions": {
    "HasCustomProviderName": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderName"}, ""]}]},
    "HasCustomRoleName": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleName"}, ""]}]},
```

```
   "HasPermissionsBoundary": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"RolePermissionsBoundary"}, ""]}]},
   "HasPolicy1": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy1"},
""]}]},
   "HasPolicy2": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy2"},
""]}]},
   "HasPolicy3": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy3"},
""]}]},
   "HasPolicy4": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy4"},
""]}]},
   "HasPolicy5": {"Fn::Not": [{"Fn::Equals": [{"Ref": "RoleManagedPolicy5"},
""]}]},
   "HasPrivateKey": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAddPrivateKey"}, ""]}]},
   "HasAssertionEncryptionMode": {"Fn::Not": [{"Fn::Equals": [{"Ref":
"IdentityProviderAssertionEncryptionMode"}, ""]}]}
 },
 "Resources": {
   "SAMLProvider": {
     "Type": "AWS::IAM::SAMLProvider",
     "Properties": {
       "Name": {"Fn::If": ["HasCustomProviderName", {"Ref":
"IdentityProviderName"}, {"Ref": "AWS::NoValue"}]},
       "SamlMetadataDocument": {"Ref": "IdentityProviderSAMLMetadataDocument"},
       "AddPrivateKey": {"Fn::If": ["HasPrivateKey", {"Ref":
"IdentityProviderAddPrivateKey"}, {"Ref": "AWS::NoValue"}]},
       "AssertionEncryptionMode": {"Fn::If": ["HasAssertionEncryptionMode", {"Ref":
"IdentityProviderAssertionEncryptionMode"}, {"Ref": "AWS::NoValue"}]}
     }
   },
   "SAMLFederatedRole": {
     "Type": "AWS::IAM::Role",
     "Properties": {
       "RoleName": {"Fn::If": ["HasCustomRoleName", {"Ref": "RoleName"}, {"Ref":
"AWS::NoValue"}]},
       "Path": {"Ref": "RolePath"},
       "Description": "SAML federated IAM role for SSO access with specified
permissions",
       "MaxSessionDuration": {"Ref": "RoleSessionDuration"},
       "PermissionsBoundary": {"Fn::If": ["HasPermissionsBoundary", {"Ref":
"RolePermissionsBoundary"}, {"Ref": "AWS::NoValue"}]},
       "AssumeRolePolicyDocument": {
         "Version": "2012-10-17",
         "Statement": [
```

```
            {
              "Effect": "Allow",
              "Principal": {
                "Federated": {"Ref": "SAMLProvider"}
              },
              "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity",
                "sts:TagSession"
              ],
              "Condition": {
                "StringEquals": {
                  "SAML:aud": "https://signin.aws.amazon.com/saml"
                }
              }
            }
          ]
        },
        "ManagedPolicyArns": {
          "Fn::Split": [
            ",",
            {
              "Fn::Join": [
                ",",
                [
                  {"Fn::If": ["HasPolicy1", {"Ref": "RoleManagedPolicy1"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy2", {"Ref": "RoleManagedPolicy2"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy3", {"Ref": "RoleManagedPolicy3"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy4", {"Ref": "RoleManagedPolicy4"}, {"Ref":
"AWS::NoValue"}]},
                  {"Fn::If": ["HasPolicy5", {"Ref": "RoleManagedPolicy5"}, {"Ref":
"AWS::NoValue"}]}
                ]
              ]
            }
          ]
        }
      }
    }
  },
  "Outputs": {
```

```
    "RoleARN": {
      "Description": "ARN of the created IAM Role",
      "Value": {"Fn::GetAtt": ["SAMLFederatedRole", "Arn"]},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-RoleARN"}
      }
    },
    "IdentityProviderARN": {
      "Description": "ARN of the created SAML Identity Provider",
      "Value": {"Ref": "SAMLProvider"},
      "Export": {
        "Name": {"Fn::Sub": "${AWS::StackName}-IdentityProviderARN"}
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] IAM: tutorial_saml-idp-and-federated-role'

Parameters:
  IdentityProviderName:
    Type: String
    Description: Name of the SAML Identity Provider (leave empty for auto-generated
 name like '{StackName}-{UniqueId}')
    Default: ""
    AllowedPattern: '^$|^[a-zA-Z0-9._-]+$'
    ConstraintDescription: Must be empty or contain only alphanumeric characters,
 periods, underscores, and hyphens

  IdentityProviderSAMLMetadataDocument:
    Type: String
    Description: SAML metadata document from identity provider

  IdentityProviderAddPrivateKey:
    Type: String
    Description: Optional private key for decrypting SAML assertions. The private
 key must be a .pem file that uses AES-GCM or AES-CBC encryption algorithm to
 decrypt SAML assertions.
    Default: ""
```

```
 IdentityProviderAssertionEncryptionMode:
   Type: String
   Description: Optional, sets encryption mode for SAML assertions
   Default: ""
   AllowedValues:
     - ""
     - "Allowed"
     - "Required"

 RoleName:
   Type: String
   Description: Name of the IAM Role (leave empty for auto-generated name like
'{StackName}-{UniqueId}')
   Default: ""
   AllowedPattern: '^$|^[\w+=,.@-]{1,64}$'
   ConstraintDescription: "Must be empty or 1-64 characters and can contain
alphanumeric characters and +=,.@-"

 RolePath:
   Type: String
   Description: Path for the IAM Role
   AllowedPattern: (^\/$)|(^\/.*\/$)
   Default: "/"

 RolePermissionsBoundary:
   Type: String
   Description: Optional ARN of the permissions boundary policy (leave empty for
none)
   Default: ""

 RoleSessionDuration:
   Description: The maximum session duration (in seconds) that you want to set for
the specified role (3600-43200)
   Type: Number
   MinValue: 3600
   MaxValue: 43200
   Default: 7200

 RoleManagedPolicy1:
   Type: String
   Description: Optional managed policy ARN 1
   Default: ""
 RoleManagedPolicy2:
   Type: String
```

```
    Description: Optional managed policy ARN 2
    Default: ""
  RoleManagedPolicy3:
    Type: String
    Description: Optional managed policy ARN 3
    Default: ""
  RoleManagedPolicy4:
    Type: String
    Description: Optional managed policy ARN 4
    Default: ""
  RoleManagedPolicy5:
    Type: String
    Description: Optional managed policy ARN 5
    Default: ""

Conditions:
  HasCustomProviderName: !Not [!Equals [!Ref IdentityProviderName, ""]]
  HasCustomRoleName: !Not [!Equals [!Ref RoleName, ""]]
  HasPermissionsBoundary: !Not [!Equals [!Ref RolePermissionsBoundary, ""]]
  HasPolicy1: !Not [!Equals [!Ref RoleManagedPolicy1, ""]]
  HasPolicy2: !Not [!Equals [!Ref RoleManagedPolicy2, ""]]
  HasPolicy3: !Not [!Equals [!Ref RoleManagedPolicy3, ""]]
  HasPolicy4: !Not [!Equals [!Ref RoleManagedPolicy4, ""]]
  HasPolicy5: !Not [!Equals [!Ref RoleManagedPolicy5, ""]]
  HasPrivateKey: !Not [!Equals [!Ref IdentityProviderAddPrivateKey, ""]]
  HasAssertionEncryptionMode: !Not [!Equals [!Ref
 IdentityProviderAssertionEncryptionMode, ""]]

Resources:
  SAMLProvider:
    Type: AWS::IAM::SAMLProvider
    Properties:
      Name: !If
        - HasCustomProviderName
        - !Ref IdentityProviderName
        - !Ref AWS::NoValue
      SamlMetadataDocument: !Ref IdentityProviderSAMLMetadataDocument
      AddPrivateKey: !If
        - HasPrivateKey
        - !Ref IdentityProviderAddPrivateKey
        - !Ref AWS::NoValue
      AssertionEncryptionMode: !If
        - HasAssertionEncryptionMode
        - !Ref IdentityProviderAssertionEncryptionMode
```

```
        - !Ref AWS::NoValue

  SAMLFederatedRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !If
        - HasCustomRoleName
        - !Ref RoleName
        - !Ref AWS::NoValue
      Path: !Ref RolePath
      Description: "SAML federated IAM role for SSO access with specified
 permissions"
      MaxSessionDuration: !Ref RoleSessionDuration
      PermissionsBoundary: !If
        - HasPermissionsBoundary
        - !Ref RolePermissionsBoundary
        - !Ref AWS::NoValue
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Federated: !Ref SAMLProvider
            Action:
              - 'sts:AssumeRole'
              - 'sts:SetSourceIdentity'
              - 'sts:TagSession'
            Condition:
              StringEquals:
                'SAML:aud': 'https://signin.aws.amazon.com/saml'
      ManagedPolicyArns: !Split
        - ','
        - !Join
          - ','
          - - !If [HasPolicy1, !Ref RoleManagedPolicy1, !Ref "AWS::NoValue"]
            - !If [HasPolicy2, !Ref RoleManagedPolicy2, !Ref "AWS::NoValue"]
            - !If [HasPolicy3, !Ref RoleManagedPolicy3, !Ref "AWS::NoValue"]
            - !If [HasPolicy4, !Ref RoleManagedPolicy4, !Ref "AWS::NoValue"]
            - !If [HasPolicy5, !Ref RoleManagedPolicy5, !Ref "AWS::NoValue"]

Outputs:
  RoleARN:
    Description: ARN of the created IAM Role
    Value: !GetAtt SAMLFederatedRole.Arn
```

```
    Export:
      Name: !Sub '${AWS::StackName}-RoleARN'

  IdentityProviderARN:
    Description: ARN of the created SAML Identity Provider
    Value: !Ref SAMLProvider
    Export:
      Name: !Sub '${AWS::StackName}-IdentityProviderARN'
```

Identidades do IAM

Uma identidade do IAM pode ser associada a uma ou mais políticas que determinam quais ações uma identidade está autorizada a realizar, em quais recursos da AWS e sob quais condições. As identidades do IAM incluem usuários do IAM, grupos do IAM e perfis do IAM. Uma entidade do IAM é um tipo de identidade que representa um usuário humano ou uma workload programática que pode ser autenticada e autorizada para executar ações na Contas da AWS. As entidades do IAM incluem usuários do IAM e perfis do IAM. Para obter as definições de termos comumente usados, consulte Termos.

Você pode federar as identidades existentes fornecidas um provedor de identidades externo. Essas identidades assumirão perfis do IAM para acessar os recursos da AWS. Para obter mais informações, consulte the section called “Federação e provedores de identidade na AWS”.

Você também pode usar o Centro de Identidade do AWS IAM para criar e gerenciar identidades e acesso aos recursos da AWS. Os conjuntos de permissões do IAM Identity Center criam automaticamente os perfis do IAM necessários para fornecer acesso aos recursos. Para obter mais informações, consulte What is IAM Identity Center?

Você pode usar o gerenciador de acesso à conta — um atributo do IAM que permite atribuir perfis do IAM existentes com o conjunto completo de atributos de perfis do IAM configurado para usuários e grupos do IAM Identity Center. Você pode usá-lo junto com os conjuntos de permissões ou sozinho.

O Usuário raiz da conta da AWS é uma entidade principal do Conta da AWS que é criada quando a sua Conta da AWS é criada. O usuário-raiz tem acesso a todos os serviços e recursos da AWS na conta. Para obter mais informações, consulte the section called “Usuário-raiz do IAM”.

> **Note**
>
> - Siga as práticas recomendadas de segurança no IAM quando trabalhar com identidades do IAM.
> - Siga as práticas recomendadas de usuário-raiz da sua Conta da AWS quando trabalhar com o usuário-raiz.
> - Se tiver problemas para iniciar uma sessão, consulte Sign in to the Console de gerenciamento da AWS.

Usuário-raiz do IAM

Ao criar uma Conta da AWS, você começa com uma identidade de login que tenha acesso completo a todos os recursos e Serviços da AWS na conta. Essa identidade é chamada deConta da AWS usuário-raiz. Para obter mais informações, consulte Usuário-raiz da conta da AWS.

Usuários do IAM

Um usuário do IAM é uma identidade na Conta da AWS que tem permissões específicas para uma única pessoa ou aplicação. Para obter mais informações, consulte Usuários do IAM.

Grupos de usuários do IAM

Um grupo de usuários do IAM é uma identidade que especifica um conjunto de usuários do IAM. Para obter mais informações, consulte Grupos de usuários.

Perfis do IAM

Um perfil do IAM é uma identidade na Conta da AWS que tem permissões específicas. Ele se assemelha a um usuário do IAM, entretanto, não está associado a uma pessoa específica. Para obter mais informações, consulte os perfis do IAM.

Usuário raiz da conta da AWS

Ao criar uma conta da Amazon Web Services (AWS) pela primeira vez, você começa com uma única identidade de login que tem acesso total a todos os produtos e recursos da AWS na conta. Essa identidade é chamada de usuário raiz da conta da AWS. O endereço de e-mail e a senha que você usou para criar sua Conta da AWS são as credenciais que você usa para fazer login como usuário-raiz.

- Use o usuário-raiz apenas para executar as tarefas que exigem permissões de nível raiz. Para obter a lista completa das tarefas que exigem fazer login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz.
- Siga as Práticas recomendadas para o usuário-raiz para sua Conta da AWS.
- Se tiver problemas para iniciar uma sessão, consulte Sign in to the Console de gerenciamento da AWS.

> **Important**
>
> É altamente recomendável não usar o usuário-raiz para tarefas diárias e seguir as melhores práticas do usuário-raiz para suas Conta da AWS. Proteja as credenciais do usuário-raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem fazer login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz.

Embora a MFA seja aplicada para usuários-raiz por padrão, ela exige uma ação do cliente para adicionar a MFA durante a criação inicial da conta, ou conforme solicitado durante o login. Para obter mais informações sobre como usar a MFA para proteger o usuário-raiz, consulte Autenticação multifator para Usuário raiz da conta da AWS.

Além do acesso ao console, você pode gerenciar o acesso programático para o usuário-raiz. Você pode Criar chaves de acesso para o usuário-raiz e Excluir chaves de acesso para o usuário-raiz quando eles não forem mais necessários.

Gerencie centralmente o acesso raiz para contas-membro

Para ajudá-lo a gerenciar credenciais em grande escala, é possível proteger centralmente o acesso às credenciais de usuário-raiz para contas-membro no AWS Organizations. Ao habilitar o AWS Organizations, você combina todas as suas contas do AWS em uma organização para gerenciamento central. A centralização do acesso raiz permite que você remova as credenciais de usuário-raiz e execute as tarefas privilegiadas a seguir nas contas-membro.

Remoção de credenciais de usuário-raiz de conta-membro

Depois de centralizar o acesso raiz para contas-membro, será possível optar por excluir as credenciais de usuário-raiz das contas-membro no seu Organizations. Você pode remover a senha do usuário-raiz, as chaves de acesso e os certificados de assinatura e desativar a autenticação multifator (MFA). As novas contas que você criar no Organizations não terão credenciais de usuário-raiz por padrão. As contas-membro não podem fazer login com o usuário-raiz nem realizar a recuperação da senha do usuário-raiz, a menos que a recuperação de conta esteja habilitada.

Realização de tarefas privilegiadas que exijam credenciais de usuário-raiz

Algumas tarefas poderão ser executadas apenas quando você fizer login como usuário-raiz de uma conta. Algumas dessas Tarefas que exigem credenciais de usuário-raiz podem ser

executadas pela conta de gerenciamento ou pelo administrador delegado do IAM. Para saber mais sobre como realizar ações privilegiadas em contas-membro, consulte Execução de uma tarefa privilegiada.

Habilitar a recuperação da conta do usuário-raiz

Se você precisar recuperar as credenciais de usuário-raiz de uma conta-membro, a conta de gerenciamento do Organizations ou o administrador delegado podem realizar a tarefa privilegiada Permitir recuperação de senha. A pessoa com acesso à caixa de entrada de e-mail do usuário-raiz da conta-membro pode redefinir a senha do usuário-raiz para recuperar as credenciais do usuário-raiz. Recomendamos excluir as credenciais de usuário-raiz após concluir a tarefa que requer acesso ao usuário-raiz.

Centralização de acesso raiz para contas-membro

As credenciais de usuário-raiz são as credenciais iniciais atribuídas a cada Conta da AWS que tenha acesso completo a todos os serviços e recursos da AWS na conta. Ao habilitar o AWS Organizations, você combina todas as suas contas do AWS em uma organização para gerenciamento central. Cada conta-membro tem seu próprio usuário-raiz com permissões padrão para realizar qualquer ação na conta-membro. Recomendamos que você proteja centralmente as credenciais de usuário-raiz das Contas da AWS gerenciadas usando o AWS Organizations para impedir a recuperação e o acesso às credenciais de usuário-raiz em grande escala.

Depois de centralizar o acesso raiz, é possível optar por excluir as credenciais de usuário-raiz das contas-membro em sua organização. Você pode remover a senha do usuário-raiz, as chaves de acesso e os certificados de assinatura e desativar a autenticação multifator (MFA). As novas contas que você criar no AWS Organizations não terão credenciais de usuário-raiz por padrão. As contas dos membros não podem fazer login com o usuário-raiz nem realizar a recuperação da senha do usuário-raiz.

> **Note**
>
> Embora algumas Tarefas que exigem credenciais de usuário-raiz possam ser executadas pela conta gerencial ou pelo administrador delegado do IAM, algumas tarefas só podem ser executadas quando você fizer login como usuário-raiz de uma conta.
> Se precisar recuperar as credenciais de usuário-raiz de uma conta de membro para realizar uma dessas tarefas, siga as etapas em Execução de uma tarefa privilegiada e selecione Permitir recuperação de senha. A pessoa com acesso à caixa de entrada de e-mail do

usuário-raiz da conta de membro poderá redefinir a senha do usuário-raiz e fazer login no usuário-raiz da conta de membro.
Recomendamos excluir as credenciais de usuário-raiz depois de concluir a tarefa que requer acesso ao usuário-raiz.

Pré-requisitos

Antes de centralizar o acesso raiz, você deve ter uma conta configurada com as configurações a seguir:

- Você deve ter as seguintes permissões do IAM:
  - `iam:GetAccessKeyLastUsed`
  - `iam:GetAccountSummary`
  - `iam:GetLoginProfile`
  - `iam:GetUser`
  - `iam:ListAccessKeys`
  - `iam:ListMFADevices`
  - `iam:ListSigningCertificates`
  - `sts:AssumeRoot`

> **Note**
>
> Para auditar o status de credencial de usuário-raiz de uma conta de membro, você pode usar a política gerenciada IAMAuditRootUserCredentials AWS para reduzir o escopo das permissões ao realizar uma tarefa privilegiada em uma conta de membro AWS Organizations, ou usar qualquer política com acesso a `iam:GetAccountSummary`. Para gerar o relatório de informações de credenciais de usuário-raiz, outras políticas só precisam da ação `iam:GetAccountSummary` para produzir a mesma saída. Você também pode listar ou obter informações individuais de credenciais de usuário-raiz, incluindo:
>
> - Se há uma senha de usuário-raiz
> - Se há uma chave de acesso de usuário-raiz e quando ela foi usada pela última vez
> - Se o usuário-raiz possui certificados de assinatura associados
> - Dispositivos de MFA associados ao usuário-raiz

- Lista do status consolidado das credenciais do usuário-raiz

- Você deve gerenciar suas Contas da AWS no AWS Organizations.
- É preciso ter as permissões a seguir para habilitar esse atributo na sua organização:
  - `iam:EnableOrganizationsRootCredentialsManagement`
  - `iam:EnableOrganizationsRootSessions`
  - `iam:ListOrganizationsFeatures`
  - `organizations:EnableAwsServiceAccess`
  - `organizations:ListAccountsForParent`
  - `organizations:RegisterDelegatedAdministrator`
- Para garantir a funcionalidade ideal do console, recomendamos habilitar as seguintes permissões adicionais:
  - `organizations:DescribeAccount`
  - `organizations:DescribeOrganization`
  - `organizations:ListAWSServiceAccessForOrganization`
  - `organizations:ListDelegatedAdministrators`
  - `organizations:ListOrganizationalUnitsForParent`
  - `organizations:ListParents`
  - `organizations:ListTagsForResource`

Habilitação do acesso raiz centralizado (console)

Para habilitar esse atributo para contas-membro no Console de gerenciamento da AWS

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Gerenciar acesso raiz e, em seguida, selecione Habilitar.

> **Note**
>
> Se você vir que o Gerenciamento de acesso raiz está desabilitado, habilite o acesso confiável para o AWS Identity and Access Management no AWS Organizations. Para

obter detalhes, consulte AWS IAM e o AWS Organizations no Guia do usuário do AWS Organizations.

3. Na seção Capacidades para habilitar, escolha quais atributos habilitar.

   - Selecione Gerenciamento de credenciais de raiz para permitir que a conta de gerenciamento e o administrador delegado do IAM excluam as credenciais de usuário-raiz das contas-membro. Você deve habilitar as Ações de usuário-raiz privilegiado nas contas-membro para permitir que as contas-membro recuperem suas credenciais de usuário-raiz após serem excluídas.
   - Selecione Ações de usuário-raiz privilegiado em contas-membro para permitir que a conta de gerenciamento e o administrador delegado do IAM realizem determinadas tarefas que exijam credenciais de usuário-raiz.

4. (Opcional) Insira o ID da conta do Administrador delegado que está autorizado a gerenciar o acesso de usuário-raiz e a realizar ações privilegiadas nas contas-membro. Recomendamos uma conta destinada a fins de segurança ou gerenciamento.

5. Escolha Habilitar.

Habilitação do acesso raiz centralizado (AWS CLI)

Para habilitar o acesso raiz centralizado a partir da AWS Command Line Interface (AWS CLI)

1. Se você ainda não habilitou o acesso confiável para o AWS Identity and Access Management no AWS Organizations, use o comando a seguir: aws organizations enable-aws-service-access.

2. Use o comando a seguir para permitir que a conta de gerenciamento e o administrador delegado do IAM excluam as credenciais de usuário-raiz das contas-membro: aws iam enable-organizations-root-credentials-management.

3. Use o comando a seguir para permitir que a conta de gerenciamento e o administrador delegado do IAM realizem tarefas que exijam credenciais de usuário-raiz: aws iam enable-organizations-root-sessions.

4. (Opcional) Use o comando a seguir para registrar um administrador delegado: aws organizations register-delegated-administrator.

   O exemplo a seguir atribui a conta 111111111111 como administrador delegado do serviço IAM.

```
aws organizations register-delegated-administrator
--service-principal iam.amazonaws.com
```

```
--account-id 111111111111
```

Habilitação do acesso raiz centralizado (AWS API)

Para habilitar o acesso raiz centralizado a partir da AWS API

1. Se você ainda não habilitou o acesso confiável para o AWS Identity and Access Management no AWS Organizations, use o comando a seguir: EnableAWSServiceAccess.
2. Use o comando a seguir para permitir que a conta de gerenciamento e o administrador delegado do IAM excluam as credenciais de usuário-raiz das contas-membro: EnableOrganizationsRootCredentialsManagement.
3. Use o comando a seguir para permitir que a conta de gerenciamento e o administrador delegado do IAM realizem tarefas que exijam credenciais de usuário-raiz: EnableOrganizationsRootSessions.
4. (Opcional) Use o comando a seguir para registrar um administrador delegado: RegisterDelegatedAdministrator.

Próximas etapas

Depois de proteger centralmente as credenciais privilegiadas para as contas-membro em sua organização, consulte Execução de uma tarefa privilegiada para realizar ações privilegiadas em uma conta-membro.

Execução de uma tarefa privilegiada em uma conta-membro do AWS Organizations

A conta gerencial da AWS Organizations ou uma conta de administrador delegado do IAM podem executar algumas tarefas privilegiadas em contas de membro que, de outra forma, exigiriam credenciais de usuário-raiz. Com o acesso raiz centralizado, essas tarefas são executadas por meio de sessões privilegiadas de curto prazo. Essas sessões fornecem credenciais temporárias com escopo para ações privilegiadas específicas, sem exigir o login do usuário-raiz na conta de membro.

Depois de iniciar uma sessão privilegiada, será possível excluir uma política de bucket do Amazon S3 mal configurada, excluir uma política de fila do Amazon SQS mal configurada, excluir as credenciais de usuário-raiz de uma conta-membro e reativar as credenciais de usuário-raiz de uma conta-membro.

> **Note**
>
> Para usar o acesso raiz centralizado, você deve fazer login por meio de uma conta gerencial ou uma conta de administrador delegado com um usuário ou perfil do IAM com a permissão `sts:AssumeRoot` explicitamente concedida. Você pode usar credenciais usuário-raiz para chamar `sts:AssumeRoot`.

Pré-requisitos

Antes de iniciar uma sessão privilegiada, é preciso ter as configurações a seguir:

- Você habilitou o acesso raiz centralizado em sua organização. Para obter etapas para ativar esse atributo, consulte Centralização de acesso raiz para contas-membro.
- Sua conta de gerenciamento ou conta de administrador delegado tem as permissões a seguir: `sts:AssumeRoot`

Realização de uma ação privilegiada em uma conta-membro (console)

Para iniciar uma sessão para ação privilegiada em uma conta-membro no Console de gerenciamento da AWS

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Gerenciamento de acesso raiz.
3. Selecione um nome na lista de contas-membro e escolha Realizar ação privilegiada.
4. Escolha a ação privilegiada que você deseja realizar na conta-membro.
    - Selecione Excluir política de bucket do Amazon S3 para remover uma política de bucket mal configurada que impeça que todas as entidades principais acessem o bucket do Amazon S3.
        a. Escolha Pesquisar no S3 para selecionar um nome dentre os buckets pertencentes à conta-membro, e selecione Escolher.
        b. Escolha Excluir política de bucket.

c. Use o console do Amazon S3 para corrigir a política de bucket após excluir a política mal configurada. Para obter mais informações, consulte Adição de uma política de bucket com o console do Amazon S3 no Guia do usuário do Amazon S3.

- Selecione Excluir política do Amazon SQS para excluir uma política baseada em recursos do Amazon Simple Queue Service que negue a todas as entidade principais acesso a uma fila do Amazon SQS.

  a. Insira o nome da fila em Nome da fila do SQS e selecione Excluir política do SQS.

  b. Use o console do Amazon SQS para corrigir a política de fila após excluir a política mal configurada. Para obter mais informações, consulte Configuração de uma política de acesso no Amazon SQS, no Guia do desenvolvedor do Amazon SQS.

- Selecione Excluir credenciais de usuário-raiz para remover o acesso raiz de uma conta-membro. A exclusão das credenciais do usuário-raiz removerá a senha do usuário-raiz, as chaves de acesso e os certificados de assinatura e desativará a autenticação multifator (MFA) da conta do membro.

  - Escolha Excluir credenciais de usuário-raiz.

- Selecione Permitir recuperação de senha para recuperar as credenciais de usuário-raiz para uma conta-membro.

  Essa opção só estará disponível quando a conta-membro não tiver credenciais de usuário-raiz.

  a. Escolha Permitir recuperação de senha.

  b. Depois de realizar essa ação privilegiada, a pessoa com acesso à caixa de entrada de e-mail do usuário-raiz da conta-membro poderá redefinir a senha do usuário-raiz e fazer login no usuário-raiz da conta-membro.

Realização de uma ação privilegiada em uma conta-membro (AWS CLI)

Para iniciar uma sessão para ação privilegiada em uma conta-membro a partir da AWS Command Line Interface

1. Use o comando a seguir para assumir uma sessão de usuário-raiz: aws sts assume-root.

> **Note**
>
> Não há suporte para o endpoint global em `sts:AssumeRoot`. Você deve enviar essa solicitação a um endpoint regional do AWS STS. Para obter mais informações, consulte Gerenciar o AWS STS em uma Região da AWS.

Ao iniciar uma sessão de usuário-raiz privilegiado para uma conta-membro, você deverá definir `task-policy-arn` estabelecer o escopo da sessão para a ação privilegiada a ser executada durante a sessão. É possível usar uma das políticas gerenciadas pela AWS a seguir para definir o escopo das ações de sessão privilegiada.

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Para limitar as ações que uma conta de gerenciamento ou administrador delegado pode realizar durante uma sessão de usuário-raiz privilegiado, é possível usar a chave de condição do AWS STS sts:TaskPolicyArn.

No exemplo a seguir, o administrador delegado assume o usuário-raiz para excluir as credenciais de usuário-raiz da conta do membro de ID *111122223333*.

```
aws sts assume-root \
  --target-principal 111122223333 \
  --task-policy-arn arn=arn:aws:iam::aws:policy/root-task/
IAMDeleteRootUserCredentials \
  --duration-seconds 900
```

2. Use `SessionToken`, `AccessKeyId` e `SecretAccessKey` da resposta para realizar ações privilegiadas na conta-membro. É possível omitir o nome de usuário e a senha na solicitação para usar como padrão a conta-membro.
   - Verificação do status das credenciais de usuário-raiz. Use os comandos a seguir para verificar o status das credenciais de usuário-raiz de uma conta-membro.

  - get-user
  - get-login-profile
  - list-access-keys
  - list-signing-certificates
  - list-mfa-devices
  - get-access-key-last-used
- Exclusão das credenciais de usuário-raiz. Use os comandos a seguir para excluir o acesso raiz. Você pode remover a senha do usuário-raiz, as chaves de acesso e os certificados de assinatura e desativar a autenticação multifator (MFA) para remover todo o acesso e a recuperação do usuário-raiz.
  - delete-login-profile
  - delete-access-key
  - delete-signing-certificate
  - deactivate-mfa-device
- Exclusão de política de bucket do Amazon S3. Use os comandos a seguir para ler, editar e excluir uma política de bucket mal configurada que impeça que todas as entidades principais acessem o bucket do Amazon S3.
  - list-buckets
  - get-bucket-policy
  - put-bucket-policy
  - delete-bucket-policy
- Exclusão de política do Amazon SQS. Use os comandos a seguir para visualizar e excluir uma política baseada em recursos do Amazon Simple Queue Service que negue a todas as entidade principais acesso a uma fila do Amazon SQS.
  - list-queues
  - get-queue-url
  - get-queue-attributes
  - set-queue-attributes
- Permissão para recuperação de senha. Use os comandos a seguir para visualizar o nome do usuário e recuperar as credenciais de usuário-raiz para uma conta-membro.

- get-login-profile
- create-login-profile

Realização de uma ação privilegiada em uma conta-membro (AWS API)

Para iniciar uma sessão para ação privilegiada em uma conta-membro a partir da AWS API

1. Use o comando a seguir para assumir uma sessão de usuário-raiz: AssumeRoot.

> **Note**
>
> Não há suporte para o endpoint global para AssumeRoot. Você deve enviar essa solicitação a um endpoint regional do AWS STS. Para obter mais informações, consulte Gerenciar o AWS STS em uma Região da AWS.

Ao iniciar uma sessão de usuário-raiz privilegiado para uma conta-membro, você deverá definir `TaskPolicyArn` estabelecer o escopo da sessão para a ação privilegiada a ser executada durante a sessão. É possível usar uma das políticas gerenciadas pela AWS a seguir para definir o escopo das ações de sessão privilegiada.

- IAMAuditRootUserCredentials
- IAMCreateRootUserPassword
- IAMDeleteRootUserCredentials
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Para limitar as ações que uma conta de gerenciamento ou administrador delegado pode realizar durante uma sessão de usuário-raiz privilegiado, é possível usar a chave de condição do AWS STS sts:TaskPolicyArn.

No exemplo a seguir, o administrador delegado assume o usuário-raiz para ler, editar e excluir uma política baseada em recursos mal configurada para um bucket do Amazon S3 para a conta-membro de ID *111122223333*.

```
https://sts.us-east-2.amazonaws.com/
```

```
?Version=2011-06-15
&Action=AssumeRoot
&TargetPrincipal=111122223333
&PolicyArns.arn=arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy
&DurationSeconds=900
```

2. Use `SessionToken`, `AccessKeyId` e `SecretAccessKey` da resposta para realizar ações privilegiadas na conta-membro. É possível omitir o nome de usuário e a senha na solicitação para usar como padrão a conta-membro.

   - Verificação do status das credenciais de usuário-raiz. Use os comandos a seguir para verificar o status das credenciais de usuário-raiz de uma conta-membro.

     - GetUser
     - GetLoginProfile
     - ListAccessKeys
     - ListSigningCertificates
     - ListMFADevices
     - GetAccessKeyLastUsed

   - Exclusão das credenciais de usuário-raiz. Use os comandos a seguir para excluir o acesso raiz. Você pode remover a senha do usuário-raiz, as chaves de acesso e os certificados de assinatura e desativar a autenticação multifator (MFA) para remover todo o acesso e a recuperação do usuário-raiz.

     - DeleteLoginProfile
     - DeleteAccessKey
     - DeleteSigningCertificate
     - DeactivateMfaDevice

   - Exclusão de política de bucket do Amazon S3. Use os comandos a seguir para ler, editar e excluir uma política de bucket mal configurada que impeça que todas as entidades principais acessem o bucket do Amazon S3.

     - ListBuckets
     - GetBucketPolicy
     - PutBucketPolicy
     - DeleteBucketPolicy

- Exclusão de política do Amazon SQS. Use os comandos a seguir para visualizar e excluir uma política baseada em recursos do Amazon Simple Queue Service que negue a todas as entidade principais acesso a uma fila do Amazon SQS.
  - ListQueues
  - GetQueueUrl
  - GetQueueAttributes
  - SetQueueAttributes
- Permissão para recuperação de senha. Use os comandos a seguir para visualizar o nome do usuário e recuperar as credenciais de usuário-raiz para uma conta-membro.
  - GetLoginProfile
  - CreateLoginProfile

Autenticação multifator para Usuário raiz da conta da AWS

> **Important**
>
> A AWS recomenda usar uma chave de acesso ou chave de segurança para MFA na AWS sempre que possível, pois elas são mais resistentes a ataques como phishing. Para obter mais informações, consulte Chaves de acesso e chaves de segurança.

A autenticação multifator (MFA) é um mecanismo simples e eficaz para aumentar sua segurança. O primeiro fator, sua senha, é um segredo que você memoriza, também conhecido como fator de conhecimento. Outros fatores podem ser fatores de posse (algo que você tem, como uma chave de segurança) ou fatores de inerência (algo que você é, como um escaneamento biométrico). Para obter mais segurança, recomendamos veementemente que você configure a autenticação multifator (MFA) para ajudar a proteger seus recursos da AWS.

> **Note**
>
> Todos os tipos de Conta da AWS (contas autônomas, de gerenciamento e de membros) exigem que a MFA seja configurada para o usuário-raiz. Os usuários precisarão registrar a MFA em até 35 dias após a primeira tentativa de login, para acessar o Console de gerenciamento da AWS, se a MFA ainda não estiver habilitada.

É possível habilitar a MFA para o Usuário raiz da conta da AWS ou para usuários do IAM. Quando você habilitar a MFA para o usuário-raiz, ela afetará somente as credenciais do usuário-raiz. Para obter mais informações sobre como habilitar a MFA para seus usuários do IAM, consulte AWSCódigo da autenticação multifator no IAM da.

> **Note**
>
> As Contas da AWS gerenciadas usando o AWS Organizations podem ter a opção de gerenciar centralmente o acesso raiz das contas de membros para evitar a recuperação de credenciais e o acesso em grande escala. Se essa opção estiver habilitada, você poderá excluir as credenciais de usuário-raiz das contas de membros, incluindo senhas e MFA, impedindo efetivamente o login como usuário-raiz, a recuperação de senha ou a configuração da MFA. Como alternativa, se você preferir manter os métodos de login baseados em senha, proteja a conta registrando a MFA para aprimorar a proteção da conta.

Antes de habilitar a MFA para seu usuário-raiz, revise e atualize as configurações e as informações de contato da sua conta para verificar se você tem acesso ao e-mail e ao número de telefone. Se o dispositivo com MFA for perdido, roubado ou não estiver funcionando, você ainda poderá fazer login como usuário-raiz verificando sua identidade usando esse e-mail e número de telefone. Para saber como fazer login usando fatores alternativos de autenticação, consulte Recuperar uma identidade protegida por MFA no IAM. Para desabilitar este recurso, entre em contato com AWS Support.

AWSA é compatível com os seguintes tipos de MFA para seu usuário-raiz:

- Chaves de acesso e chaves de segurança
- Aplicações de autenticador virtual
- Tokens físicos de TOTP

Chaves de acesso e chaves de segurança

AWS Identity and Access ManagementA é compatível com chaves de acesso e chaves de segurança para MFA. Com base nos padrões FIDO, chaves de acesso usam criptografia de chave pública para proporcionar uma autenticação forte e resistente a phishing que é mais segura do que as senhas. A AWS é compatível com dois tipos de chaves de acesso: chaves de acesso vinculadas a dispositivo (chaves de segurança) e chaves de acesso sincronizadas.

- Chaves de segurança: dispositivos físicos, como a YubiKey, usados como segundo fator de autenticação. Uma única chave de segurança é compatível com várias contas de usuário-raiz e usuários do IAM.
- Chaves de acesso sincronizadas: usam gerenciadores de credenciais de provedores como Google, Apple, contas da Microsoft e serviços de terceiros, como 1Password, Dashlane e Bitwarden, como segundo fator.

É possível usar autenticadores biométricos integrados, como Touch ID em Apple MacBooks, para desbloquear seu gerenciador de credenciais e fazer login na AWS. As chaves de acesso são criadas com o provedor escolhido usando sua impressão digital, rosto ou PIN do dispositivo. É possível usar uma chave de acesso de autenticação entre dispositivos (CDA) de um dispositivo, como um dispositivo móvel ou uma chave de segurança de hardware, para entrar em outro dispositivo, como um laptop. Para obter mais informações, consulte autenticação entre dispositivos (CDA).

É possível sincronizar chaves de acesso em seus dispositivos para facilitar o login na AWS e aprimorar a usabilidade e a capacidade de recuperação. Para obter mais informações sobre como habilitar chaves de acesso e chaves de segurança, consulte Habilitar uma chave de acesso ou uma chave de segurança para o usuário-raiz (console).

A FIDO Alliance conta com uma lista de todos os produtos certificados pela FIDO compatíveis com as especificações da FIDO.

Aplicações de autenticador virtual

Uma aplicação de autenticador virtual funciona em um telefone ou outro dispositivo e emula um dispositivo físico. As aplicações de autenticação virtual implementam o algoritmo de senha de uso único com marcação temporal (TOTP) e oferecem suporte a vários tokens em um único dispositivo. O usuário deve digitar um código válido no dispositivo quando solicitado durante o processo de acesso. Cada token atribuído a um usuário deve ser exclusivo. O usuário não pode digitar um código do token de outro usuário para se autenticar.

Recomendamos que você use um dispositivo MFA virtual ao mesmo tempo em que aguarda a aprovação da compra do hardware ou enquanto aguarda a chegada do hardware. Para obter uma lista de algumas aplicações compatíveis que podem ser usadas como dispositivos de MFA virtuais, consulte Autenticação multifator (MFA). Para obter instruções sobre como configurar um dispositivo MFA virtual com a AWS, consulte Habilitar um dispositivo MFA virtual para o usuário-raiz (console).

Tokens físicos de TOTP

Um dispositivo físico gera um código numérico de seis dígitos baseado no algoritmo de senha de uso único com marcação temporal (TOTP). O usuário deve digitar um código válido no dispositivo em uma segunda página da web durante o login. Cada dispositivo MFA atribuído a um usuário deve ser exclusivo. Um usuário não pode digitar um código do dispositivo de outro usuário para ser autenticado. Para obter informações sobre os dispositivos físicos de MFA compatíveis, consulte autenticação multifator (MFA). Para obter instruções sobre como configurar um token de hardware TOTP com a AWS, consulte Habilitar um token de hardware TOTP para o usuário-raiz (console).

Se você quiser usar um dispositivo físico de MFA, recomendamos que use chaves de segurança FIDO como alternativa aos dispositivos físicos de TOTP. As chaves de segurança FIDO proporcionam a vantagem de não usar bateria, ter resistência a phishing e comportar vários usuários-raiz e usuários do IAM em um único dispositivo para maior segurança.

Tópicos

- Habilitar uma chave de acesso ou uma chave de segurança para o usuário-raiz (console)
- Habilitar um dispositivo MFA virtual para o usuário-raiz (console)
- Habilitar um token de hardware TOTP para o usuário-raiz (console)

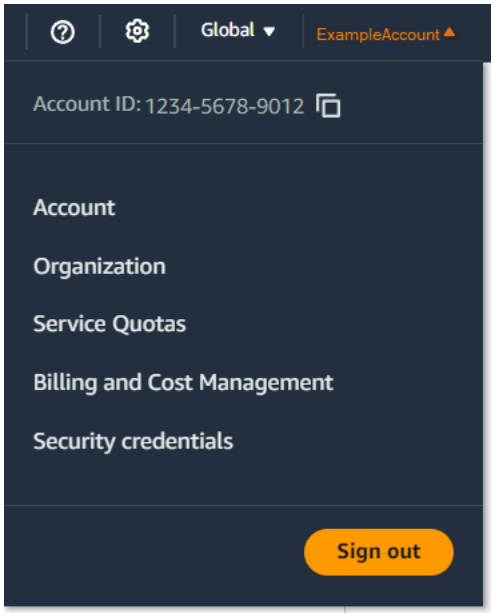
Habilitar uma chave de acesso ou uma chave de segurança para o usuário-raiz (console)

Você só pode configurar e habilitar uma chave de acesso para o seu usuário-raiz no Console de gerenciamento da AWS, não na AWS CLI e nem na API da AWS.

Para habilitar uma chave de acesso ou uma chave de segurança para seu usuário-raiz do IAM (console)

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.
2. No lado direito da barra de navegação, selecione o nome da conta e escolha Security credentials (Credenciais de segurança).



3. Na página Minhas credenciais de segurança do usuário-raiz, em Autenticação multifator (MFA), escolha Atribuir dispositivo de MFA.
4. Na página de nome do dispositivo de MFA, insira o nome do dispositivo, escolha Chave de acesso ou Chave de segurança e, em seguida, escolha Avançar.
5. Em Configurar dispositivo, configure sua chave de acesso. Crie uma chave de acesso com dados biométricos, como seu rosto ou impressão digital, com um pin do dispositivo ou inserindo a chave de segurança FIDO na porta USB do seu computador e tocando nela.
6. Siga as instruções no seu navegador para escolher um provedor de chave de acesso ou onde você deseja armazená-la para uso em todos os seus dispositivos.
7. Escolha Continuar.

Agora, você registrou sua chave de acesso para uso com a AWS. A próxima vez que usar as credenciais de usuário-raiz para fazer login, você deverá se autenticar com sua chave de acesso para concluir o processo de login.

Para obter ajuda na solução de problemas de chaves de segurança FIDO, consulte Solucionar problemas de chaves de acesso e chaves de segurança FIDO.

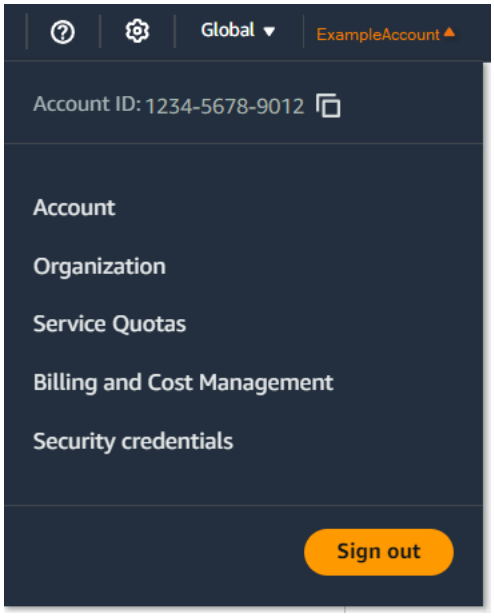
Habilitar um dispositivo MFA virtual para o usuário-raiz (console)

É possível usar o Console de gerenciamento da AWS para configurar e habilitar um dispositivo com MFA virtual para seu usuário-raiz. Para habilitar dispositivos com MFA para a Conta da AWS, você deve estar conectado à AWS usando suas credenciais de usuário-raiz.

Para configurar e habilitar um dispositivo com MFA virtual para uso com seu usuário-raiz (console)

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.
2. No lado direito da barra de navegação, escolha o nome de sua conta e escolha Security credentials (Credenciais de segurança).



3. Na seção Multi-Factor Authentication (MFA) (Autenticação multifator [MFA]), selecione Assign MFA device (Atribuir dispositivo de MFA).
4. No assistente, digite um Nome de dispositivo, escolha Aplicação de autenticador e escolha Próximo.

   O IAM gera e exibe informações de configuração para o dispositivo com MFA virtual, incluindo um código QR gráfico. O gráfico é uma representação da chave de configuração secreta que está disponível para entrada manual em dispositivos que não suportam códigos de QR.
5. Abra o aplicativo MFA virtual no dispositivo.

   Se o aplicativo de MFA virtual oferecer suporte a vários dispositivos ou contas de MFA virtual, selecione a opção para criar uma conta ou um dispositivo MFA virtual.
6. A maneira mais fácil de configurar o aplicativo é usar o aplicativo para digitalizar o código QR. Se você não puder digitalizar o código, é possível digitar as informações de configuração manualmente. O código QR e a chave de configuração secreta gerados pelo IAM estão vinculados a sua Conta da AWS e não podem ser usados com outra conta. Entretanto, eles

podem ser reutilizados para configurar um novo dispositivo MFA para a sua conta caso você perca acesso ao dispositivo MFA original.

- Para usar o código de QR para configurar o dispositivo MFA virtual, no assistente, escolha Mostrar código de QR. Em seguida, siga as instruções do app para digitalizar o código. Por exemplo, pode ser necessário selecionar o ícone de câmera ou escolher um comando como Scan account barcode (Digitalizar código de barras da conta) e usar a câmera do dispositivo para digitalizar o código QR.
- No assistente Set up device (Configurar dispositivo), selecione Show secret key (Exibir chave secreta) e digite a chave secreta em sua aplicação de MFA.

> **Important**
>
> Faça um backup seguro do código QR ou da chave de configuração secreta, ou habilite vários dispositivos de MFA para sua conta. É possível registrar até oito dispositivos com MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu Usuário raiz da conta da AWS e usuários do IAM. Um dispositivo de MFA virtual pode ficar indisponível, por exemplo, se você perder o smartphone no qual o dispositivo de MFA virtual está hospedado. Se isso acontecer e você não conseguir fazer login em sua conta sem dispositivos com MFA adicionais anexados ao usuário ou até por Recuperar um dispositivo com MFA de usuário raiz, não poderá fazer login em sua conta e terá de entrar em contato com o atendimento ao cliente para remover a proteção de MFA da conta.

O dispositivo começa a gerar números de seis dígitos.

7. No assistente, na caixa MFA code 1 (Código MFA 1), digite a senha de uso único que atualmente é exibida no dispositivo de MFA virtual. Espere até 30 segundos para que o dispositivo gere uma nova senha de uso único. Em seguida, digite a segunda senha de uso único na caixa Código MFA 2. Escolha Add MFA (Adicionar MFA).

> **Important**
>
> Envie sua solicitação imediatamente após gerar o código. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA associa com êxito ao usuário, mas o dispositivo MFA está fora de sincronia. Isso ocorre porque as senhas

únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Caso isso ocorra, você pode ressincronizar o dispositivo.

O dispositivo está pronto para uso com a AWS. Para obter informações sobre como usar a MFA com o Console de gerenciamento da AWS, consulte Login habilitado para MFA.

Habilitar um token de hardware TOTP para o usuário-raiz (console)

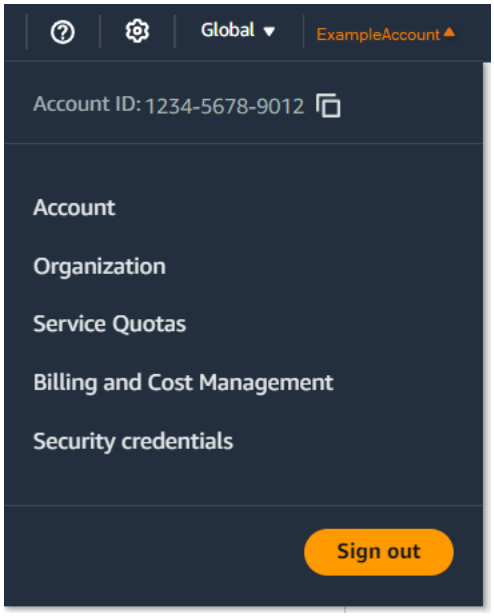
Você só pode configurar e habilitar um dispositivo de MFA físico para o seu usuário-raiz no Console de gerenciamento da AWS, não na AWS CLI e nem na API da AWS.

> **Note**
>
> É possível ver um texto diferente, como Fazer login usando MFA e Solucionar problemas do dispositivo de autenticação. No entanto, os mesmos recursos são fornecidos. Em ambos os casos, se você não puder verificar o endereço de e-mail e o número de telefone de sua conta usando fatores alternativos de autenticação, entre em contato com o AWS Support para excluir sua configuração de MFA.

Para habilitar um token de hardware TOTP para o usuário-raiz (console)

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.
2. No lado direito da barra de navegação, selecione o nome da conta e escolha Security credentials (Credenciais de segurança).



3. Expanda a seção Multi-factor authentication (MFA) (Autenticação multifator (MFA)).
4. Escolha Assign MFA device (Atribuir dispositivo de MFA).
5. No assistente, digite um Device name (Nome de dispositivo), escolha Hardware TOTP token (Token de hardware TOTP) e escolha Next (Avançar).
6. Na caixa Serial number (Número de série), insira o número de série localizado na parte de trás do dispositivo MFA.
7. Na caixa MFA code 1 (Código MFA 1), digite o número de seis dígitos exibido pelo dispositivo MFA. Talvez seja necessário pressionar o botão na parte frontal do dispositivo para exibir o número.

8. Aguarde 30 segundos enquanto o dispositivo atualiza o código e digite o próximo número de seis dígitos na caixa MFA code 2 (Código MFA 2). Talvez seja necessário pressionar o botão na parte frontal do dispositivo novamente para exibir o segundo número.
9. Escolha Add MFA (Adicionar MFA). O dispositivo de MFA agora está associado à Conta da AWS.

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos de autenticação. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA será

associado com êxito ao usuário, mas o dispositivo MFA ficará fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Caso isso ocorra, você pode ressincronizar o dispositivo.

A próxima vez que você usar suas credenciais de usuário-raiz para fazer login, você deve digitar um código do dispositivo MFA.

Alterar a senha para o Usuário raiz da conta da AWS

É possível alterar o endereço de e-mail e a senha na página Credenciais de segurança ou Conta. Você também pode selecionar Esqueceu a senha? na página de login da AWS para redefinir sua senha.

Para alterar a senha do usuário-raiz., é necessário fazer login como Usuário raiz da conta da AWS, e não como um usuário do IAM. Para saber como redefinir uma senha de usuário-raiz esquecida, consulte Redefinir uma senha de usuário-raiz perdida ou esquecida.

Para proteger sua senha, é importante seguir estas melhores práticas:

- Altere sua senha periodicamente.
- Mantenha sua senha privada, pois qualquer pessoa que souber sua senha poderá acessar sua conta.
- Use na AWS uma senha diferente da que você usa em outros sites.
- Evite senhas que sejam fáceis de adivinhar. Elas incluem senhas como `secret`, `password`, `amazon` ou `123456`. Evite também usar termos como palavras do dicionário, o seu nome, endereço de e-mail ou outras informações pessoais que possam ser facilmente obtidas.

Important

As Contas da AWS gerenciadas por meio do AWS Organizations podem ter acesso raiz centralizado habilitado para as contas de membro. Essas contas-membro não têm credenciais de usuário-raiz, não podem fazer login como usuário-raiz e são impedidas de recuperar a senha do usuário-raiz. Entre em contato com seu administrador se precisar realizar uma tarefa que exija credenciais de usuário-raiz.

Console de gerenciamento da AWS

Para alterar a senha para o usuário-raiz

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:
>
> - Você deve fazer login como usuário-raiz da Conta da AWS, o que não requer permissões adicionais do AWS Identity and Access Management (IAM). Não é possível executar essas etapas como usuário ou perfil do IAM.

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.
2. No canto superior direito do console, selecione o nome de sua conta e as Credenciais de segurança.
3. Na página Account (Conta), ao lado de Account settings (Configurações da conta), escolha Edit (Editar). Você será avisado para se autenticar novamente para fins de segurança.

   > **Note**
   >
   > Se você não vir a opção Editar, é provável que você não esteja conectado como usuário-raiz da sua conta. Não será possível modificar as configurações da conta enquanto estiver conectado como usuário ou perfil do IAM.
4. Na página Atualizar configurações da conta, em Senha, escolha Editar.
5. Na página Atualizar sua senha, preencha os campos Senha atual, Nova senha e Confirmar nova senha.

   > **Important**
   >
   > Certifique-se de escolher uma senha forte. Embora você possa definir uma política de senha de conta para usuários do IAM, essa política não se aplica ao usuário-raiz.

AWSA exige que sua senha atenda às seguintes condições:

- Ter no mínimo 8 caracteres e no máximo 128 caracteres de extensão.
- Incluir no mínimo três dos seguintes tipos de caracteres: maiúsculas, minúsculas, números e os símbolos ! @ # $ % ^ & * () <> [] {} | _+-=.
- Não ser idêntica ao nome ou endereço de e-mail da sua Conta da AWS.

6. Escolha Salvar alterações.

AWS CLI or AWS SDK

Não há suporte a essa tarefa na AWS CLI ou por uma operação de API de um dos AWS SDKs. É possível realizar essa tarefa somente usando o Console de gerenciamento da AWS.

Redefinir uma senha de usuário-raiz perdida ou esquecida

Quando você criou sua Conta da AWS inicialmente, forneceu um endereço de e-mail e uma senha. Essas são as suas credenciais de Usuário raiz da conta da AWS. Se você esquecer sua senha de usuário-raiz, poderá redefinir a senha no Console de gerenciamento da AWS.

As Contas da AWS gerenciadas por meio do AWS Organizations podem ter acesso raiz centralizado habilitado para as contas de membro. Essas contas-membro não têm credenciais de usuário-raiz, não podem fazer login como usuário-raiz e são impedidas de recuperar a senha do usuário-raiz. Entre em contato com seu administrador se precisar realizar uma tarefa que exija credenciais de usuário-raiz.

Important

Está com problemas para fazer login na AWS? Certifique-se de estar na página de login da AWS correta para o seu tipo de usuário. Se você for o Usuário raiz da conta da AWS (proprietário da conta), poderá fazer login na AWS usando as credenciais que configurou ao criar a Conta da AWS. Se você é um usuário do IAM, o administrador da conta poderá fornecer as credenciais que você pode usar para fazer login na AWS. Se você precisar solicitar suporte, não use o link de feedback nesta página, pois o formulário é recebido pela equipe de documentação da AWS, não pelo Suporte. Em vez disso, na página Entre em

contato conosco, escolha Ainda não consegue fazer login em sua conta da AWS e escolha uma das opções de suporte disponíveis.

Como redefinir sua senha de usuário-raiz

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.

   > **Note**
   >
   > Se você estiver conectado ao Console de gerenciamento da AWS com as credenciais de usuário do IAM, deverá sair do sistema para que possa redefinir a senha de usuário-raiz. Se você vir a página de login do usuário do IAM específica da conta, escolha Sign-in using root account credentials (Fazer login usando as credenciais da conta raiz) perto da parte inferior da página. Se necessário, forneça o endereço de e-mail da conta e selecione Next (Próximo) para acessar a página Root user sign in (Login do usuário-raiz).

2. Escolha Esqueceu sua senha?.

   > **Note**
   >
   > Se você for um usuário do IAM, essa opção não estará disponível. A opção Esqueceu sua senha? está disponível somente para a conta do usuário-raiz. Os usuários do IAM devem solicitar que o administrador redefina uma senha esquecida. Para obter mais informações, consulte Esqueci a senha de usuário do IAM da minha conta da AWS. Se você iniciar sessão via portal de acesso do AWS, consulte Redefinir a senha de usuário do IAM Identity Center.

3. Forneça o endereço de e-mail associado à conta. Em seguida, forneça o texto CAPTCHA e escolha Continuar.
4. Verifique o e-mail associado à sua Conta da AWS para ver se há uma mensagem da Amazon Web Services. O e-mail será enviado por um endereço que termina em `@verify.signin.aws`. Siga as orientações no e-mail. Se você não vir o e-mail em sua conta, verifique a pasta de spam.

Se você não tiver mais acesso ao e-mail, consulte Não tenho acesso ao e-mail de minha conta da AWS no Guia do usuário do Início de Sessão da AWS.

Criar chaves de acesso para o usuário-raiz

> **Warning**
>
> É altamente recomendável não criar pares de chaves de acesso para o usuário raiz. Como apenas algumas tarefas exigem o usuário raiz e elas normalmente são executadas com pouca frequência, recomendamos entrar no Console de gerenciamento da AWS para realizar as tarefas de usuário raiz. Antes de criar chaves de acesso, avalie as alternativas às chaves de acesso de longo prazo.

Embora não seja recomendável, é possível criar chaves de acesso para seu usuário raiz para executar comandos na AWS Command Line Interface (AWS CLI) ou usar operações de API de um dos AWS SDKs usando credenciais de usuário raiz. Quando você cria chaves de acesso, o ID da chave de acesso e a chave de acesso secreta são criados como um conjunto. Durante a criação de chaves de acesso, a AWS oferece a você uma oportunidade de visualizar e fazer download da chave de acesso secreta que faz parte de uma chave de acesso. Se não fizer o download ou perdê-lo, você pode excluir a chave de acesso e, em seguida, criar uma nova. É possível criar chaves de acesso do usuário raiz com o console, a AWS CLI ou a API da AWS.

Uma chave de acesso recém-criada tem o status de ativo, o que significa que você pode usar a chave de acesso para chamadas de API e CLI. É possível atribuir até duas chaves de acesso ao usuário raiz.

As chaves de acesso que não estiverem em uso deverão ser desativadas. Depois que uma chave de acesso estiver inativa, você não poderá usá-la para chamadas de API. As chaves inativas ainda contam para o seu limite. Você pode criar ou excluir uma chave de acesso a qualquer momento. No entanto, se uma chave de acesso for excluída, isto será definitivo, e não poderá ser recuperada.

Console de gerenciamento da AWS

Para criar uma chave de acesso para o Usuário raiz da conta da AWS

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:

- Você deve fazer login como usuário-raiz da Conta da AWS, o que não requer permissões adicionais do AWS Identity and Access Management (IAM). Não é possível executar essas etapas como usuário ou perfil do IAM.

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.
2. No canto superior direito do console, selecione o nome de sua conta e, em seguida, selecione Credenciais de segurança.
3. Na seção Chaves de acesso, escolha Criar chave de acesso. Se essa opção não estiver disponível, então você já tem o número máximo de chaves de acesso. Você deverá excluir uma das chaves de acesso existentes antes de poder criar outra chave. Para obter mais informações, consulte Cotas de objetos do IAM.
4. Na página Alternativas às chaves de acesso do usuário raiz, revise as recomendações de segurança. Para continuar, marque a caixa de seleção e escolha Criar chave de acesso.
5. Na página Recuperar chave de acesso, sua ID da chave de acesso será exibida.
6. Em Chave de acesso secreta, selecione Mostrar e copie o ID da chave de acesso e a chave secreta da janela do navegador para colá-la em outro lugar seguro. Como alternativa, é possível escolher Baixar arquivo. csv, o que iniciará o download de um arquivo denominado `rootkey.csv` que contém o ID da chave de acesso e a chave secreta. Salve o arquivo em um lugar seguro.
7. Selecione Concluído. Quando você não precisar mais usar a chave de acesso, recomendamos que a exclua ou, pelo menos, considere desativá-la, para que não seja mal utilizada.

AWS CLI & SDKs

Para criar uma chave de acesso para o usuário raiz

> **Note**
>
> Para executar o comando ou operação de API a seguir como usuário raiz, você já deve ter um par de chaves de acesso ativo. Se você ainda não tiver nenhuma chave de

acesso, crie a primeira chave de acesso usando o Console de gerenciamento da AWS. Em seguida, será possível usar as credenciais dessa primeira chave de acesso com a AWS CLI para criar a segunda chave de acesso ou excluir uma chave de acesso.

- AWS CLI: aws iam create-access-key

  Example

```
$ aws iam create-access-key
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- API da AWS: CreateAccessKey na Referência da API do IAM.

Excluir chaves de acesso para o usuário-raiz

É possível usar o Console de gerenciamento da AWS, a AWS CLI ou a API da AWS para excluir as chaves de acesso do usuário raiz.

Console de gerenciamento da AWS

Para excluir uma chave de acesso para o usuário raiz

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:
>
> - Você deve fazer login como usuário-raiz da Conta da AWS, o que não requer permissões adicionais do AWS Identity and Access Management (IAM). Não é possível executar essas etapas como usuário ou perfil do IAM.

1. Abra o AWS Management Console e inicie a sessão usando suas credenciais de usuário-raiz.

   Para obter instruções, consulte Sign in to the Console de gerenciamento da AWS as the root user no Início de Sessão da AWS User Guide.
2. No canto superior direito do console, selecione o nome de sua conta e, em seguida, selecione Credenciais de segurança.
3. Na seção Chaves de acesso, selecione a chave de acesso que deseja excluir e, em seguida, em Ações escolha Excluir.

   > **Note**
   >
   > Como alternativa, é possível Desativar uma chave de acesso, em vez de excluí-la permanentemente. Dessa forma é possível retomar o uso dela futuramente, sem a necessidade de alterar o ID da chave ou a chave secreta. Enquanto a chave estiver inativa, todas as tentativas de usá-la em solicitações para a API da AWS falharão com o status de acesso negado.
4. Na caixa de diálogo Excluir <ID da chave de acesso>, escolha Desativar, insira o ID da chave de acesso para confirmar que você deseja excluí-la e escolha Excluir.

AWS CLI & SDKs

Para excluir uma chave de acesso para o usuário raiz

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:
>
> - Você deve fazer login como usuário-raiz da Conta da AWS, o que não requer permissões adicionais do AWS Identity and Access Management (IAM). Não é possível executar essas etapas como usuário ou perfil do IAM.

- AWS CLI: aws iam delete-access-key

Example

```
$ aws iam delete-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE
```

Esse comando não gera nenhuma saída quando é bem-sucedido.

- API da AWS: DeleteAccessKey

Tarefas que exigem credenciais de usuário-raiz

Recomendamos que configurar um usuário administrativo no Centro de Identidade do AWS IAM para realizar tarefas diárias e acessar os recursos da AWS. Porém, as tarefas listadas abaixo podem ser executadas apenas quando você fizer login como o usuário-raiz de uma conta.

Para simplificar o gerenciamento de credenciais de usuário-raiz privilegiado em todas as contas-membro no AWS Organizations, é possível habilitar o acesso raiz centralizado para ajudá-lo a proteger centralmente o acesso altamente privilegiado às suas Contas da AWS. Gerencie centralmente o acesso raiz para contas-membropermite que você remova centralmente e evite a recuperação de credenciais de usuário-raiz a longo prazo, melhorando a segurança da conta em sua organização. Depois que você habilitar esse atributo, poderá executar as tarefas privilegiadas a seguir nas contas-membro.

- Remova as credenciais do usuário-raiz da conta-membro para evitar a recuperação da conta do usuário-raiz. Também é possível permitir a recuperação de senha para recuperar as credenciais de usuário-raiz para uma conta-membro.
- Remova uma política de bucket mal configurada que negue a todas as entidades principais o acesso ao bucket do Amazon S3.
- Exclua uma política baseada em recursos do Amazon Simple Queue Service que negue a todas as entidade principais acesso a uma fila do Amazon SQS.

Tarefas de gerenciamento de contas

- Alterar as configurações da sua Conta da AWS. Contas da AWS autônomas que não fazem parte do AWS Organizations requerem credenciais de usuário-raiz para atualização do endereço de e-mail, da senha do usuário-raiz e das chaves de acesso do usuário-raiz. Outras configurações da

conta, como nome da conta, informações de contato, contatos alternativos, preferência de moeda de pagamento e Regiões da AWS, não requerem credenciais de usuário-raiz.

> **Note**
>
> AWS OrganizationsO , com todos os atributos habilitados, pode ser usado para gerenciar as configurações das contas de membros centralmente, na conta gerencial e nas contas de administrador delegado. Usuários ou perfis do IAM autorizados, tanto na conta gerencial quanto nas contas de administrador delegado, podem fechar as contas de membros e atualizar os endereços de e-mail, os nomes das contas, as informações de contato, os contatos alternativos e as Regiões da AWS das contas de membros.

- Feche sua Conta da AWS. Contas da AWS autônomas que não fazem parte do AWS Organizations requerem credenciais de usuário-raiz para serem fechadas. Com o AWS Organizations, você pode fechar as contas de membros centralmente, na conta gerencial e nas contas de administrador delegado.
- Reabra sua Conta da AWS. Se você mesmo fechou uma conta independente e ela não tem saldo pendente, você pode reabri-la durante o período pós-encerramento. Você deve fazer login como usuário-raiz para fazer isso.
- Restaurar permissões do usuário do IAM. Se o único administrador do IAM revogar acidentalmente suas próprias permissões, será possível fazer login como o usuário-raiz para editar políticas e restaurar essas permissões.

Tarefas de cobrança

- Ativação do acesso do IAM ao console de Gerenciamento de Faturamento e Custos.
- Algumas tarefas de cobrança são limitadas ao usuário-raiz. Consulte Gerenciando uma Conta da AWS no Guia de usuário do AWS Billing para obter mais informações.
- Exiba determinadas faturas de imposto. Um usuário do IAM com a permissão aws-portal:ViewBilling pode visualizar e fazer download de faturas de IVA da AWS Europa, mas não da AWS Inc. ou da Amazon Internet Services Private Limited (AISPL).

AWS GovCloud (US)Tarefas do

- Cadastre-se no AWS GovCloud (US).
- Solicitar as chaves de acesso do usuário-raiz da conta AWS GovCloud (US) ao AWS Support.

Tarefa do Amazon EC2

- Registrado como um vendedor no Marketplace de instâncias reservadas.

AWS KMSTarefa do

- Caso uma chave do AWS Key Management Service perca o controle, um administrador poderá recuperá-la entrando em contato com o Suporte. No entanto, o Suporte responderá ao número de telefone principal do usuário-raiz para autorização confirmando a OTP do ticket.

Tarefas do Amazon Simple Storage Service

- Configurar um bucket do Amazon S3 para habilitar a MFA (autenticação multifator).
- Editar ou excluir uma política de bucket do Amazon S3 que negue todas as entidade principais.

  Você pode usar ações privilegiadas para desbloquear um bucket do Amazon S3 com uma política de bucket mal configurada. Para obter detalhes, consulte Execução de uma tarefa privilegiada em uma conta-membro do AWS Organizations.

Tarefa do Amazon Simple Queue Service

- Edite ou exclua uma política baseada em recurso do Amazon SQS que negue todas as entidade principais.

  Você pode usar ações privilegiadas para desbloquear uma fila do Amazon SQS com uma política baseada em recursos mal configurada. Para obter detalhes, consulte Execução de uma tarefa privilegiada em uma conta-membro do AWS Organizations.

Recursos adicionais

Para obter mais informações sobre o usuário-raiz da AWS, consulte os recursos a seguir:

- Para obter ajuda com problemas do usuário-raiz, consulte Solucionar problemas com o usuário-raiz.
- Para gerenciar centralmente os endereços de e-mail do usuário-raiz no AWS Organizations, consulte Como atualizar o endereço de e-mail do usuário-raiz de uma conta do membro no Guia do usuário do AWS Organizations.

Os artigos a seguir fornecem informações adicionais sobre como trabalhar com o usuário-raiz.

- Quais são algumas das práticas recomendadas para proteger minha Conta da AWS e seus recursos?
- Como posso criar uma regra de evento do EventBridge para me notificar se meu usuário-raiz for usado?
- Monitorar e notificar atividades do Usuário raiz da conta da AWS
- Monitorar a atividade do usuário-raiz do IAM

Usuários do IAM

> **Important**
>
> As práticas recomendadas do IAM aconselham exigir que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias em vez de usuários do IAM com credenciais de longo prazo. Recomendamos usar somente usuários do IAM para casos de uso específicos não compatíveis com usuários federados.
> Se você estiver usando nossa nova experiência da AWS e quiser fornecer aos usuários humanos acesso aos seus projetos ou as Configurações da AWS, adicione-os à sua equipe. Em nossa nova experiência, você não pode criar usuários do IAM com acesso ao console. Para obter mais informações, consulte Convidar membros da equipe.

Um usuário do IAM é uma entidade que você cria na sua Conta da AWS. O usuário do IAM representa a pessoa ou a workload que usa o usuário do IAM para interagir com a AWS. Um usuário do IAM consiste em um nome e credenciais.

Um usuário do IAM com permissões de administrador não é o mesmo que um Usuário raiz da conta da AWS. Para obter mais informações sobre o usuário raiz, consulte Usuário raiz da conta da AWS.

Como a AWS identifica um usuário do IAM

Quando você cria um usuário do IAM, o IAM cria as seguintes maneiras de identificar esse usuário:

- Um "nome amigável" para o usuário do IAM, que é o nome que você especificou ao criar esse usuário, como `Richard` ou `Anaya`. Esses são os nomes que você vê no Console de

gerenciamento da AWS. Como os nomes de usuário do IAM aparecem nos nomes dos recursos da Amazon (ARNs), não recomendamos incluir informações de identificação pessoal no nome do IAM. Consulte Requisitos de nome do IAM para obter requisitos e restrições de nomes do IAM.

- Um nome de recurso da Amazon (ARN) para o usuário do IAM. Você usa o ARN quando precisa identificar exclusivamente esse usuário do IAM em toda a AWS. Por exemplo, você pode usar um ARN para especificar o usuário do IAM como uma `Principal` em uma política do IAM para um bucket do Amazon S3. Um ARN para um usuário do IAM pode se parecer com o seguinte:

  ```
  arn:aws:iam::account-ID-without-hyphens:user/Richard
  ```

- Um identificador exclusivo para o usuário do IAM. Esse ID é retornado somente quando você usa a API, o Tools for Windows PowerShell ou a AWS CLI para criar o usuário do IAM. Ele não está visível no console.

Para obter mais informações sobre esses identificadores, consulte Identificadores do IAM.

Usuários do IAM e credenciais

Você pode acessar a AWS de diferentes maneiras dependendo das credenciais do usuário do IAM:

- Senha do console: uma senha que o usuário do IAM pode digitar para fazer login em sessões interativas, como o Console de gerenciamento da AWS. Desabilitar a senha (acesso ao console) de um usuário do IAM impede que ele faça login no Console de gerenciamento da AWS usando suas credenciais de login. Isso não altera as permissões do usuário nem o impede de acessar o console usando uma função assumida. Os usuários do IAM com acesso ao console habilitado também podem usar essas mesmas credenciais para autenticar a AWS CLI e acessar o SDK usando o comando `aws login` da AWS CLI. Esses usuários precisarão ter permissões SignInLocalDevelopmentAccess. Para obter mais detalhes, consulte Autenticação e credenciais de acesso para a AWS CLI no Guia do usuário da AWS Command Line Interface.
- Chaves de acesso: usadas para fazer chamadas programáticas para a AWS. Porém, existem alternativas mais seguras a serem consideradas antes de criar chaves de acesso para usuários do IAM. Para obter mais informações, consulte Considerações e alternativas para chaves de acesso de longo prazo na Referência geral da AWS. Se o usuário do IAM tiver chaves de acesso ativas, elas continuarão funcionando e permitirão o acesso por meio da AWS CLI, do Tools for Windows PowerShell, da API da AWS ou do AWS Console Mobile Application.
- Chaves SSH para uso com CodeCommit: uma chave SSH pública no formato OpenSSH que pode ser usada para autenticação com CodeCommit.

- Certificados de servidor: Certificados SSL/TLS que você pode usar para autenticar com alguns serviços da AWS. Recomendamos que você use o Gerenciador de certificados da AWS (ACM) para provisionar, gerenciar e implantar seus certificados de servidor. Use o IAM apenas quando precisar oferecer suporte a conexões HTTPS em uma região que não seja compatível com o ACM. Para saber quais regiões são compatíveis com o ACM, consulte Cotas e endpoints do Gerenciador de certificados da AWS na Referência geral da AWS.

Você pode escolher as credenciais certas para o seu usuário do IAM. Quando você usa o Console de gerenciamento da AWS para criar um usuário do IAM, é necessário pelo menos incluir uma senha ou chaves de acesso para o console. Por padrão, um novo usuário do IAM criado usando a AWS CLI ou a API da AWS não tem nenhum tipo de credencial. Você deve criar o tipo de credencial para um usuário do IAM com base na seu caso de uso.

Você tem as seguintes opções para administrar senhas, chaves de acesso e dispositivos com senhas, chaves de acesso e dispositivos de autenticação multifator (MFA):

- Gerenciar senhas para seus usuários do IAM. Crie e altere as senhas que permitem acesso ao Console de gerenciamento da AWS. Definir uma política de senha para impor uma complexidade mínima para a senha. Permita que os usuários do IAM alterem suas próprias senhas.
- Gerenciar chaves de acesso para seus usuários do IAM. Criar e atualizar as chaves de acesso para acesso programático aos recursos na sua conta.
- Habilite a autenticação multifator (MFA) para o usuário do IAM. Como prática recomendada, recomendamos exigir a autenticação multifator de todos os usuários do IAM na sua conta. Com a MFA, os usuários do IAM devem fornecer duas formas de identificação: primeiro, eles fornecem as credenciais que fazem parte de sua identidade de usuário (uma senha ou uma chave de acesso). Além disso, eles fornecem um código numérico temporário que é gerado em um dispositivo de hardware ou por uma aplicação em um smartphone ou tablet.
- Localizar senhas e chaves de acesso não utilizadas. Qualquer pessoa que tenha uma senha ou chaves de acesso da sua conta ou de um usuário do IAM em sua conta terá acesso aos seus recursos da AWS. A prática recomendada de segurança é remover as senhas e as chaves de acesso quando os usuários do IAM não precisarem mais delas.
- Fazer download de um relatório de credenciais para sua conta. Você pode gerar e baixar um relatório de credenciais que lista todos os usuários do IAM em sua conta e o status de diversas credenciais deles, incluindo senhas, chaves de acesso e dispositivos com MFA. Em caso de senhas e chaves de acesso, o relatório de credenciais mostra quando uma senha ou chave de acesso foi usada recentemente.

Usuários do IAM e permissões

Por padrão, um novo usuário do IAM não tem permissões para fazer nada. Ele não está autorizado a executar nenhuma operação da AWS ou a acessar qualquer recurso da AWS. Uma vantagem de ter usuários do IAM individuais é que você pode atribuir permissões individualmente para cada usuário. Você pode atribuir permissões administrativas para alguns usuários que, por sua vez, podem administrar seus recursos da AWS e até mesmo criar e gerenciar outros usuários do IAM. Na maioria dos casos, entretanto, você deseja limitar as permissões de um usuário apenas às tarefas (ações ou operações da AWS) e aos recursos necessários para o trabalho.

Imagine um usuário chamado Diego. Ao criar o usuário do IAM `Diego`, você cria uma senha para ele e anexa permissões que permitem que ele execute determinada instância do Amazon EC2 e leia (`GET`) informações de uma tabela em um banco de dados do Amazon RDS. Para conhecer os procedimentos para criar usuários do IAM e conceder as credenciais e as permissões iniciais, consulte Criar um usuário do IAM na Conta da AWS. Para obter os procedimentos sobre como alterar as permissões para usuários existentes, consulte Alterar permissões de um usuário do IAM. Para obter os procedimentos sobre como alterar a senha ou chaves de acesso do usuário, consulte Senhas de usuários na AWS e Gerenciar chaves de acesso para usuários do IAM.

Você também pode adicionar um limite de permissões para seus usuários do IAM. Um limite de permissões é um recurso avançado que permite usar políticas gerenciadas da AWS para limitar as permissões máximas que uma política com base em identidade pode conceder a um usuário ou perfil do IAM. Para obter mais informações sobre os tipos e os usos de políticas, consulte Políticas e permissões no AWS Identity and Access Management.

Usuários do IAM e contas

Cada usuário do IAM está associado a uma única Conta da AWS. Como os usuários da IAM são definidos na sua Conta da AWS, não precisam ter um método de pagamento registrado na AWS. Qualquer atividade da AWS executada pelos usuários do IAM na sua conta é cobrada na sua conta.

O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Usuários do IAM como contas de serviço

Um usuário do IAM é um recurso no IAM que tem credenciais e permissões associadas. Um usuário do IAM pode representar uma pessoa ou uma aplicação que usa credenciais para fazer solicitações

da AWS. Isso é geralmente chamado de conta de serviço. Se você optar por usar as credenciais de longo prazo de um usuário do IAM em sua aplicação, não incorpore chaves de acesso diretamente no código da aplicação. Os SDKs da AWS a AWS Command Line Interface permitem que você coloque as chaves de acesso em locais conhecidos, para que não seja necessário mantê-las em código. Para obter mais informações, consulte Gerenciar chaves de acesso de usuários do IAM corretamente na Referência geral da AWS. Como alternativa, e como uma prática recomendada, você pode usar credenciais de segurança temporárias (funções do IAM), em vez de chaves de acesso de longo prazo.

Depois de criar usuários do IAM, você pode Visualizar usuários do IAM em sua conta e Renomear um usuário do IAM conforme necessário.

Como os usuários do IAM fazem login na AWS

Para fazer login no Console de gerenciamento da AWS como usuário do IAM, você deve fornecer seu ID de conta ou alias de conta, além de seu nome de usuário e senha. Quando o administrador criou seu usuário do IAM no console, ele deve ter enviado suas credenciais de login, incluindo seu nome de usuário e o URL para a página de login da sua conta, que inclui o ID ou o alias da sua conta.

```
https://My_AWS_Account_ID.signin.aws.amazon.com/console/
```

> **Dica**
>
> Para criar um marcador para a página de login da conta no navegador da web, digite manualmente o URL de login da conta na entrada do marcador. Não use o recurso de marcador do navegador da web, pois os redirecionamentos podem ocultar o URL de login.

Você também pode fazer login no seguinte endpoint geral de login e digitar o ID ou o alias da conta manualmente:

```
https://console.aws.amazon.com/
```

Para maior conveniência, a página de login da AWS usa um cookie de navegador para lembrar o nome de usuário e as informações da conta do IAM. Na próxima vez que o usuário acessar qualquer página no Console de gerenciamento da AWS, o console usará o cookie para redirecionar o usuário para a página de login da conta.

Você tem acesso apenas aos recursos da AWS que seu administrador especifica na política anexada à sua identidade de usuário do IAM. Para trabalhar no console, você deve ter permissões para executar as ações que o console executa, como listar e criar recursos da AWS. Para obter mais informações, consulte Gerenciamento de acesso para recursos da AWS e Exemplos de políticas baseadas em identidade do IAM.

> **Note**
>
> Se sua organização tiver um sistema de identidade existente, você poderá criar uma opção de autenticação única (SSO). A SSO fornece aos usuários acesso ao Console de gerenciamento da AWS de sua conta sem exigir que eles tenham uma identidade de usuário do IAM. O SSO também elimina a necessidade do login dos usuários no site da organização e na AWS separadamente. Para obter mais informações, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.

Registrar detalhes de login no CloudTrail

Se você habilitar o CloudTrail para registrar eventos de login em seus logs, você deverá estar ciente de como o CloudTrail escolhe onde os eventos devem ser registrados.

- Se fizerem login diretamente em um console, os usuários serão redirecionados para um endpoint de login global ou regional, com base no suporte a regiões pelo console de serviço selecionado. Por exemplo, a página inicial do console principal é compatível com regiões, logo, se você fizer login no seguinte URL:

  ```
  https://alias.signin.aws.amazon.com/console
  ```

  você será redirecionado para um endpoint de login regional como `https://us-east-2.signin.aws.amazon.com`, o que resulta em um registro em log do CloudTrail no log da região do usuário:

  Por outro lado, o console do Amazon S3 não oferece suporte a regiões, logo, se você fizer login no URL a seguir

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

A AWS redirecionará você para o endpoint de login global em `https://signin.aws.amazon.com`, o que resulta em um registro em log do CloudTrail global.

- Você pode solicitar manualmente um determinado endpoint de login regional fazendo login na página inicial do console compatível com a região usando uma sintaxe de URL como a seguinte:

```
https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
```

A AWS redirecionará você para o endpoint de login regional `ap-southeast-1` e isso resultará em um evento de log do CloudTrail regional.

Para obter mais informações sobre o CloudTrail e o IAM, consulte Registrar eventos do IAM no CloudTrail.

Caso os usuários precisem de acesso programático para trabalhar com a conta, você pode criar um par de chaves de acesso (uma ID de chave de acesso e uma chave de acesso secreta) para cada usuário. Porém, existem alternativas mais seguras a serem consideradas antes de criar chaves de acesso para os usuários. Para obter mais informações, consulte Considerações e alternativas para chaves de acesso de longo prazo na Referência geral da AWS.

Recursos adicionais

Os recursos a seguir podem ajudar você a saber mais sobre o login da AWS.

- O Guia de usuário do Início de Sessão da AWS ajuda a entender as diferentes maneiras pelas quais você pode fazer login na Amazon Web Services (AWS), dependendo do tipo de usuário que você é.
- Você pode fazer login com até cinco identidades diferentes simultaneamente em um único navegador da Web no Console de gerenciamento da AWS. Para obter detalhes, consulte Signing in to multiple accounts no Guia de conceitos básicos do Console de gerenciamento da AWS.
- Para encontrar o ID da conta ou o alias necessário para fazer login, consulte Visualizar o ID da Conta da AWS e Usar um alias para seu ID da Conta da AWS.

Login habilitado para MFA

Os usuários configurados com dispositivos de autenticação multifator (MFA) devem usar esses dispositivos para fazer login no Console de gerenciamento da AWS. Depois que o usuário insere suas credenciais de login, a AWS verifica a conta do usuário para ver se a MFA é necessária.

> **Important**
>
> Se você usar credenciais da chave secreta e chave de acesso para acesso direto ao Console de gerenciamento da AWS com a chamada de API AWS STS`GetFederationToken`, o MFA NÃO será necessário. Para ter mais informações, consulte Usar credenciais da chave secreta e chave de acesso para acesso ao console.

Os tópicos a seguir fornecem informações sobre como os usuários concluem o login quando a MFA é necessária.

Tópicos

- Vários dispositivos com MFA habilitados
- Chave de segurança FIDO
- Dispositivo com MFA virtual
- Token físico de TOTP

Vários dispositivos com MFA habilitados

Ao fazer login no Console de gerenciamento da AWS como usuário raiz de uma Conta da AWS ou usuário do IAM com vários dispositivos de MFA habilitados para essa conta, o usuário só precisará usar um dispositivo de MFA para fazer login. Depois de fazer a autenticação com a senha do usuário, o usuário seleciona o tipo de dispositivo de MFA que gostaria de usar para concluir a autenticação. Em seguida, solicita-se que o usuário faça a autenticação com o tipo de dispositivo selecionado.

Chave de segurança FIDO

Se a MFA for necessária para o usuário, uma segunda página de login será exibida. O usuário precisa tocar na chave de segurança FIDO.

> **Note**
>
> Os usuários do Google Chrome não devem escolher nenhuma das opções disponíveis no pop-up que solicite Verify your identity with amazon.com (Confirmar sua identidade na amazon.com). Você só precisa tocar na chave de segurança.

Ao contrário de outros dispositivos de MFA, as chaves de segurança FIDO não perdem a sincronia. Os administradores podem desativar uma chave de segurança FIDO se ela for perdida ou quebrada. Para ter mais informações, consulte Desativar dispositivos MFA (console).

Para obter informações sobre os navegadores compatíveis com WebAuthn e FIDO compatíveis com a AWS, consulte Configurações compatíveis com o uso de chaves de acesso e chaves de segurança.

Dispositivo com MFA virtual

Se a MFA for necessária para o usuário, uma segunda página de login será exibida. Na caixa código MFA, o usuário deve informar o código numérico fornecido pelo aplicativo de MFA.

Se o código de MFA estiver correto, o usuário poderá acessar o Console de gerenciamento da AWS. Se o código estiver incorreto, o usuário poderá tentar novamente com outro código.

Um dispositivo MFA virtual pode estar fora de sincronia. Se após várias tentativas sem êxito um usuário não conseguir fazer login no Console de gerenciamento da AWS, será solicitado que ele sincronize o dispositivo MFA virtual. O usuário pode seguir as instruções na tela para sincronizar o dispositivo MFA virtual. Para obter informações sobre como sincronizar um dispositivo em nome de um usuário na Conta da AWS, consulte Sincronizar novamente os dispositivos MFA virtuais e de hardware.

Token físico de TOTP

Se a MFA for necessária para o usuário, uma segunda página de login será exibida. Na caixa MFA code (Código de MFA), o usuário deve informar o código numérico fornecido por um token de hardware TOTP.

Se o código de MFA estiver correto, o usuário poderá acessar o Console de gerenciamento da AWS. Se o código estiver incorreto, o usuário poderá tentar novamente com outro código.

Um token de hardware TOTP pode perder a sincronia. Se, após várias tentativas, o usuário não conseguir fazer login no Console de gerenciamento da AWS, será solicitado que ele sincronize o

dispositivo de token MFA. O usuário pode seguir as instruções na tela para sincronizar o dispositivo de token de MFA. Para obter informações sobre como sincronizar um dispositivo em nome de um usuário na Conta da AWS, consulte Sincronizar novamente os dispositivos MFA virtuais e de hardware.

Criar um usuário do IAM na Conta da AWS

> **Important**
>
> As práticas recomendadas do IAM aconselham exigir que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias em vez de usuários do IAM com credenciais de longo prazo. Recomendamos usar somente usuários do IAM para casos de uso específicos não compatíveis com usuários federados.
> Se você estiver usando nossa nova experiência da AWS e quiser fornecer aos usuários humanos acesso aos seus projetos ou as Configurações da AWS, adicione-os à sua equipe. Em nossa nova experiência, você não pode criar usuários do IAM com acesso ao console. Para obter mais informações, consulte Convidar membros da equipe.

O processo de criar um usuário do IAM e habilitá-lo para executar tarefas consiste nas seguintes etapas:

1. Crie o usuário no Console de gerenciamento da AWS, na AWS CLI, no Tools for Windows PowerShell ou usando uma operação da API da AWS. Se você criar o usuário no Console de gerenciamento da AWS, as etapas de 1 a 4 serão tratadas automaticamente com base em suas opções. Se você criar os usuários do IAM de forma programática, deverá executar cada uma dessas etapas separadamente.
2. Crie credenciais para o usuário, dependendo do tipo de acesso que o usuário requer:
   - Habilitar acesso ao console: opcional: se o usuário precisar acessar o Console de gerenciamento da AWS, crie uma senha para o usuário. Desabilitar o acesso ao console para um usuário impede que ele faça login no Console de gerenciamento da AWS usando seu próprio nome de usuário e senha. Isso não altera as permissões do usuário nem o impede de acessar o console usando uma função assumida.

> **Tip**
>
> Crie apenas as credenciais necessárias para o usuário. Por exemplo, para um usuário que precise de acesso apenas pelo Console de gerenciamento da AWS, não crie chaves de acesso.

3. Conceda ao usuário permissões para realizar as tarefas necessárias. Recomendamos que você inclua os usuários do IAM nos grupos e gerencie as permissões por meio de políticas anexadas a esses grupos. No entanto, também é possível conceder permissões associando políticas de permissões diretamente ao usuário. Se você usar o console para adicionar o usuário, poderá copiar as permissões de um usuário existente para o novo usuário.

   Você também pode adicionar um limite de permissões para limitar as permissões do usuário especificando uma política que define as permissões máximas que ele pode ter. Os limites de permissões não concedem nenhuma permissão.

   Para obter instruções sobre como criar uma política de permissão personalizada a ser usada para conceder permissões ou para definir um limite de permissões, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.
4. (Opcional) Adicione metadados ao usuário anexando tags. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
5. Forneça ao usuário as informações de login necessárias. Isso inclui a senha e a URL do console na página de login da conta em que o usuário fornece essas credenciais. Para obter mais informações, consulte Como os usuários do IAM fazem login na AWS.
6. (Opcional) Configure a multi-factor authentication (MFA) para o usuário. A MFA requer que o usuário forneça um código de uso único a cada vez que ele faz login no Console de gerenciamento da AWS.
7. (Opcional) Dê aos usuários do IAM permissões para gerenciar suas próprias credenciais de segurança. (Por padrão, os usuários do IAM não têm permissões para gerenciar suas próprias credenciais.) Para obter mais informações, consulte Permitir que os usuários do IAM alterem as próprias senhas.

> **Note**
>
> Se você usar o console para criar o usuário e selecionar O usuário deverá criar uma nova senha no próximo login (recomendado), o usuário receberá as permissões necessárias.

Para obter informações sobre as permissões de que você precisa para criar um usuário, consulte Permissões necessárias para acessar recursos do IAM.

Para obter instruções sobre como criar usuários do IAM para casos de uso específicos, consulte os seguintes tópicos:

- Criar um acesso de emergência para um usuário do IAM
- Crie um usuário do IAM para workloads que não podem usar perfis do IAM

Visualizar usuários do IAM

Você pode listar os usuários do IAM da sua Conta da AWS ou de um determinado grupo do IAM, e listar todos os grupos do IAM a que um usuário pertence. Para obter informações sobre as permissões que você precisa para listar usuários, consulte Permissões necessárias para acessar recursos do IAM.

Para listar todos os usuários do IAM na conta

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.

O console exibe os usuários do IAM da sua Conta da AWS.

AWS CLI

Execute este comando:

- `aws iam list-users`

API

Chame a seguinte operação:

- `ListUsers`

Para listar os usuários do IAM em um grupo do IAM

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione User groups (Grupos de usuários).
4. Escolha o nome do grupo de usuários.

Os usuários do IAM que são membros do grupo são listados na guia Usuários.

AWS CLI

Execute este comando:

- `aws iam get-group`

API

Chame a seguinte operação:

- `GetGroup`

Para listar todos os grupos do IAM a que um usuário pertence

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome de usuário do IAM.
5. Selecione a guia Grupos para exibir a lista de grupos que incluem o usuário atual.

AWS CLI

Execute este comando:

- `aws iam list-groups-for-user`

API

Chame a seguinte operação:

- `ListGroupsForUser`

Próximas etapas

Depois de ter uma lista dos usuários do IAM, você pode renomear, excluir ou desativar um usuário do IAM usando os procedimentos a seguir.

- Renomear um usuário do IAM
- Remover ou desativar um usuário do IAM

Renomear um usuário do IAM

> **Note**
>
> Como prática recomendada, aconselhamos exigir que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias. Seguindo as práticas recomendadas, você não gerenciará usuários e grupos do IAM. Em vez disso, seus usuários e grupos serão gerenciados fora da AWS e podem acessar recursos da AWS como identidade federada. Identidade federada é um usuário de seu diretório de usuários corporativos, um provedor de identidades da Web, AWS Directory Service, o diretório do Centro de Identidade ou qualquer usuário que acesse os serviços da AWS usando credenciais fornecidas por meio de uma fonte de identidade. As identidades federadas utilizam os grupos definidos pelo provedor de identidade. Se você estiver usando o Centro de Identidade do AWS IAM, consulte Manage identities in IAM Identity Center (Gerenciar identidades no Centro de Identidade do IAM) no Guia do usuário do Centro de Identidade do AWS IAM para obter informações sobre a criação de usuários e grupos no Centro de Identidade do IAM.

A Amazon Web Services oferece várias ferramentas para gerenciar os usuários do IAM na sua Conta da AWS. Você pode listar os usuários do IAM da sua conta ou de um grupo de usuários, ou listar todos os grupos do IAM a que um usuário pertence. Você pode renomear ou alterar o caminho de um usuário do IAM. Se estiver migrando para utilizar identidades federadas em vez de usuários do IAM, você poderá excluir um usuário do IAM da conta da AWS ou desativá-lo.

Para obter mais informações sobre como adicionar, alterar ou remover políticas gerenciadas para um usuário do IAM, consulte Alterar permissões de um usuário do IAM. Para obter informações sobre como gerenciar políticas em linha para usuários do IAM, consulte Adicionar e remover permissões de identidade do IAM, Editar políticas do IAM e Excluir políticas do IAM. Como prática recomendada, use políticas gerenciadas em vez de políticas em linha. As políticas gerenciadas pela AWS concedem permissões para vários casos de uso comuns. Lembre-se de que as políticas gerenciadas pela AWS podem não conceder permissões de privilégio mínimo para seus casos de uso específicos porque estão disponíveis para uso por todos os clientes da AWS. Como resultado, recomendamos que você reduza ainda mais as permissões definindo as políticas gerenciadas pelo cliente específicas para seus casos de uso. Para ter mais informações, consulte AWSPolíticas gerenciadas pela. Para obter mais informações sobre políticas gerenciadas pela AWS que são

projetadas para funções de trabalho específicas, consulte AWSPolíticas gerenciadas pela para funções de trabalho.

Para saber como validar políticas do IAM, consulte Validação de política do IAM.

> **Tip**
>
> O IAM Access Analyzer pode analisar os serviços e as ações que seus perfis do IAM usam e, em seguida, gerar uma política aperfeiçoada que você pode utilizar. Depois de testar cada política gerada, você pode implantar a política em seu ambiente de produção. Isso garante que você conceda apenas as permissões necessárias para suas workloads. Para obter mais informações sobre a geração de políticas, consulte Geração de políticas do IAM Access Analyzer.

Para obter informações sobre como gerenciar senhas de usuário do IAM, consulte Gerenciar senhas para usuários do IAM.

Renomeação de um usuário do IAM

Para alterar o nome ou caminho de um usuário, você deve usar a AWS CLI, o Tools for Windows PowerShell ou a API da AWS. Não há opção no console para renomear um usuário. Para obter informações sobre as permissões que você precisa para renomear um usuário, consulte Permissões necessárias para acessar recursos do IAM.

Quando você altera o nome ou caminho de um usuário, acontece o seguinte:

- Todas as políticas anexadas ao usuário permanecem com o usuário sob o novo nome.
- O usuário permanece nos mesmos grupos do IAM com o novo nome.
- O ID exclusivo para o usuário permanece o mesmo. Para obter mais informações sobre IDs exclusivos, consulte Identificadores exclusivos.
- Todas as políticas de recurso ou função que se refiram ao usuário como um principal (o usuário está sendo recebendo acesso) são automaticamente atualizadas para usar o novo nome ou caminho. Por exemplo, quaisquer políticas baseadas em fila no Amazon SQS ou políticas baseadas em recursos no Amazon S3 são atualizadas automaticamente para usar o novo nome e caminho.

O IAM não atualiza automaticamente as políticas que se referem ao usuário como um recurso para usar o novo nome ou caminho; você deve fazer isso manualmente. Por exemplo, imagine que o usuário `Richard` tenha uma política anexada que lhe permite gerenciar suas credenciais de segurança. Se um administrador renomear `Richard` como `Rich`, o administrador também precisa atualizar essa política para alterar o recurso disso:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

para isso:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

Isso também se aplica se um administrador mudar o caminho; o administrador precisa atualizar a política para refletir o novo caminho para o usuário.

Para renomear um usuário

- AWS CLI: aws iam update-user
- API da AWS: UpdateUser

Remover ou desativar um usuário do IAM

As práticas recomendadas aconselham que você remova usuários do IAM não utilizados da Conta da AWS. Se quiser reter as credenciais dos usuários do IAM para uso futuro, em vez de excluí-las da conta, você poderá desativar o acesso do usuário. Para obter mais informações, consulte Desativar um usuário do IAM.

> **Warning**
>
> Depois que um usuário do IAM e suas chaves de acesso são excluídos, não podem ser restaurados ou recuperados.

Pré-requisito: visualizar o acesso do usuário do IAM

Antes de remover um usuário, analise sua atividade recente em nível de serviço. Esse processo ajuda a evitar a remoção do acesso de uma entidade principal (pessoa ou aplicação) que o esteja

usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Remoção de um usuário do IAM (console)

Quando você usa o Console de gerenciamento da AWS para remover um usuário do IAM, o IAM exclui automaticamente as seguintes informações associadas:

- O identificador de usuário do IAM
- Quaisquer associações de grupo: ou seja, o usuário do IAM é removido de todos os grupos dos quais o usuário do IAM era membro
- Qualquer senha associada ao usuário do IAM
- Todas as políticas em linha incorporadas no usuário do IAM (as políticas que foram aplicadas ao usuário do IAM usando permissões de grupo de usuários não são afetadas)

> **Note**
>
> O IAM remove todas as políticas gerenciadas anexadas ao usuário do IAM quando você exclui o usuário, mas não exclui as políticas gerenciadas.

- Todos os dispositivos MFA associados

Para remover um usuário do IAM (console)

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Usuários e, em seguida, marque a caixa de seleção ao lado do nome de usuário do IAM que você deseja excluir.
4. Na parte superior da página, escolha Delete (Excluir).

> **Note**
>
> Se algum dos usuários tiver chaves de acesso ativas, será necessário desativar as chaves de acesso antes de excluir os usuários. Para obter mais informações, consulte Para desativar uma chave de acesso para um usuário do IAM.

5. Na caixa de diálogo de confirmação, insira o nome de usuário no campo de entrada de texto para confirmar a exclusão do usuário. Escolha Excluir.

O console exibe uma notificação de status informando que o usuário do IAM foi excluído.

Exclusão de um usuário do IAM (AWS CLI)

Ao contrário do Console de gerenciamento da AWS, quando você exclui um usuário do IAM com a AWS CLI, é necessário excluir manualmente os itens anexados ao usuário do IAM. Este procedimento ilustra o processo.

Para excluir um usuário do IAM da Conta da AWS (AWS CLI)

1. Exclua a senha do usuário, caso ele tenha uma.

   `aws iam delete-login-profile`

2. Exclui as chaves de acesso do usuário, se o usuário as tiver.

   `aws iam list-access-keys` (para listar as chaves de acesso do usuário) e `aws iam delete-access-key`

3. Exclui o certificado de assinatura do usuário. Observe que ao excluir uma credencial de segurança, ela é removida permanentemente e não pode ser recuperada.

   `aws iam list-signing-certificates` (para listar o certificados de assinatura do usuário) e `aws iam delete-signing-certificate`

4. Exclui a chave pública SSH do usuário, se o usuário tiver uma.

   `aws iam list-ssh-public-keys` (para listar as chaves públicas SSH do usuário) e `aws iam delete-ssh-public-key`

5. Exclui as credenciais do Git do usuário.

`aws iam list-service-specific-credentials` (para listar as credenciais do git do usuário) e `aws iam delete-service-specific-credential`

6. Desativa o dispositivo de autenticação multifator (MFA), se o usuário tiver um.

   `aws iam list-mfa-devices` (para listar os dispositivos MFA do usuário), `aws iam deactivate-mfa-device` (para desativar o dispositivo) e `aws iam delete-virtual-mfa-device` (para excluir permanentemente um dispositivo MFA virtual)

7. Exclui as políticas em linha do usuário.

   `aws iam list-user-policies` (para listar as políticas em linha do usuário) e `aws iam delete-user-policy` (para excluir a política)

8. Desanexa todas as políticas gerenciadas anexadas ao usuário.

   `aws iam list-attached-user-policies` (para listar as políticas gerenciadas anexadas ao usuário) e `aws iam detach-user-policy` (para desanexar a política)

9. Remova o usuário de todos os grupos do IAM.

   `aws iam list-groups-for-user` (para listar os grupos do IAM aos quais o usuário pertence) e `aws iam remove-user-from-group`

10. Exclua o usuário.

    `aws iam delete-user`

Desativar um usuário do IAM

Talvez seja necessário desativar um usuário do IAM enquanto ele estiver temporariamente fora da empresa. Você pode manter as credenciais de usuário do IAM do usuário como estão e apenas bloquear o acesso dele à AWS.

Para desativar um usuário, crie e anexe uma política que negue ao usuário acesso à AWS. Você pode restaurar o acesso do usuário posteriormente.

Aqui estão dois exemplos de políticas de negação que você pode anexar a um usuário para negar seu acesso.

A política a seguir não inclui um limite de tempo. Você deve remover a política para restaurar o acesso do usuário.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

A política a seguir inclui uma condição que inicia a política em 24 de dezembro de 2024 às 23:59 (UTC) e termina em 28 de fevereiro de 2025 às 23:59 (UTC).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

Controlar o acesso do usuário do IAM ao Console de gerenciamento da AWS

Os usuários do IAM com permissão que fizerem login na sua Conta da AWS por meio do Console de gerenciamento da AWS poderão acessar seus recursos da AWS. A lista a seguir mostra como conceder aos usuários do IAM acesso a recursos da sua Conta da AWS por meio do Console de gerenciamento da AWS. Ela também mostra como os usuários do IAM podem acessar outros recursos da AWS por meio do site da AWS.

> **Note**
>
> Não há custo pelo uso do IAM.

A Console de gerenciamento da AWS

Você cria uma senha para cada usuário do IAM que precisa acessar o Console de gerenciamento da AWS. Os usuários acessam o console por meio da página de login da sua Conta da AWS habilitada para o IAM. Para obter informações sobre como acessar a página de login, consulte Como fazer login na AWS, no Guia do usuário do Início de Sessão da AWS. Para obter informações sobre como criar senhas, leia Senhas de usuários na AWS.

Você pode impedir que um usuário do IAM acesse o Console de gerenciamento da AWS removendo a senha dele. Isso impede que ele faça login no Console de gerenciamento da AWS usando suas credenciais de login. Isso não altera as permissões do usuário nem o impede de acessar o console usando uma função assumida. Se o usuário tiver chaves de acesso ativas, elas continuarão funcionando e permitirão o acesso por meio da AWS CLI, do Tools for Windows PowerShell, da API da AWS ou do AWS Console Mobile Application.

Seus recursos da AWS, como instâncias do Amazon EC2, buckets do Amazon S3 etc.

Mesmo que os usuários do IAM tenham senhas, eles ainda precisarão de permissão para acessar seus recursos da AWS. Quando você cria um usuário do IAM, ele não tem nenhuma permissão por padrão. Para dar a seus usuários do IAM as permissões que eles precisam ter, você anexa políticas a eles. Se houver muitos usuários do IAM que executam as mesmas tarefas com os mesmos recursos, você poderá lhes atribuir a um grupo. Então, atribua as permissões a esse grupo. Para obter informações sobre a criação de usuários e grupos do IAM, consulte Identidades do IAM. Para obter informações sobre o uso de políticas para definir permissões, consulte Gerenciamento de acesso para recursos da AWS.

AWSFóruns de discussão da

Qualquer um pode ler as postagens no Fóruns de discussão da AWS. Os usuários que desejam publicar dúvidas ou comentários no Fórum de discussão da AWS podem fazê-lo usando seu nome de usuário. Na primeira vez que um usuário publica no Fórum de discussão da AWS, o usuário é solicitado a inserir um apelido e um endereço de e-mail. Somente esse usuário pode usar esse apelido nos Fóruns de discussão da AWS.

Informações de uso e faturamento da sua Conta da AWS

Você pode conceder aos usuários acesso às informações de uso e faturamento da sua Conta da AWS. Para obter mais informações, consulte Controlar o acesso às suas informações de faturamento no Guia do usuário do AWS Billing.

As informações de perfil da sua Conta da AWS

Os usuários não podem acessar as informações de perfil da sua Conta da AWS.

Credenciais de segurança da sua Conta da AWS

Os usuários não podem acessar as credenciais de segurança da sua Conta da AWS.

> **Note**
>
> As políticas do IAM controlam o acesso, independentemente da interface. Por exemplo, você pode fornecer a um usuário uma senha para acessar o Console de gerenciamento da AWS. As políticas para esse usuário (ou qualquer grupo ao qual o usuário pertence) controlam o que o usuário pode fazer no Console de gerenciamento da AWS. Ou, você pode fornecer ao usuário chaves de acesso da AWS para fazer chamadas de API para a AWS. As políticas controlam as ações que o usuário pode chamar por meio de uma biblioteca ou cliente que usa essas chaves de acesso para autenticação.

Alterar permissões de um usuário do IAM

Você pode alterar as permissões de um usuário do IAM na sua Conta da AWS alterando suas associações a grupos, copiando permissões de um usuário existente, anexando políticas diretamente a um usuário ou definindo um limite de permissões. Um limite de permissões controla o número máximo de permissões que um usuário pode ter. Os limites de permissões são um recurso avançado da AWS.

Para obter informações sobre as permissões que você precisa para modificar as permissões de um usuário, consulte Permissões necessárias para acessar recursos do IAM.

Tópicos

- Visualizar acesso do usuário
- Gerar uma política com base na atividade de acesso de um usuário
- Adição de permissões a um usuário (console)
- Alteração das permissões de um usuário (console)
- Para remover uma política de permissões de um usuário (console)
- Para remover o limite de permissões de um usuário (console)
- Adição e remoção das permissões de um usuário (AWS CLI ou API da AWS)

Visualizar acesso do usuário

Antes de alterar as permissões de um usuário, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Gerar uma política com base na atividade de acesso de um usuário

Às vezes, você pode conceder permissões a uma entidade do IAM (usuário ou função) além do que é exigido. Para ajudar você a refinar as permissões concedidas, você pode gerar uma política do IAM baseada na atividade de acesso para uma entidade. O IAM Access Analyzer revisa seus logs do AWS CloudTrail e gera um modelo de política que contém as permissões que foram usadas pela entidade no intervalo de datas especificado. Você pode usar o modelo para criar uma política gerenciada com permissões refinadas e anexá-la à entidade do IAM. Dessa forma, você concede apenas as permissões de que o usuário ou a função precisa para interagir com os recursos da AWS para seu caso de uso específico. Para saber mais, consulte Geração de políticas do IAM Access Analyzer.

Adição de permissões a um usuário (console)

O IAM oferece três formas de adicionar políticas de permissões a um usuário:

- Adicionar o usuário do IAM a um grupo do IAM: tornar o usuário membro de um grupo. As políticas do grupo são anexadas ao usuário.
- Copiar permissões de um usuário do IAM existente: copiar todas as associações de grupos, políticas gerenciadas anexadas, políticas em linha e quaisquer limites de permissões existentes do usuário de origem.
- Anexar políticas diretamente ao usuário do IAM: anexar uma política gerenciada diretamente ao usuário. Para facilitar o gerenciamento das permissões, anexe suas políticas a um grupo e associe os usuários do IAM aos grupos apropriados.

> **Important**
>
> Se o usuário tiver um limite de permissões, você não poderá adicionar mais permissões ao usuário do que as permitidas pelo limite de permissões.

Para adicionar permissões adicionando o usuário do IAM a um grupo

Adicionar um usuário do IAM a um grupo do IAM atualiza imediatamente as permissões do usuário com as permissões definidas para o grupo.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome de usuário do IAM.
5. Selecione a guia Grupos para exibir a lista de grupos que incluem o usuário atual.
6. Escolha Adicionar usuário aos grupos.
7. Marque a caixa de seleção de cada grupo no qual você deseja que o usuário ingresse. A lista mostra o nome de cada grupo e as políticas que o usuário receberá se tornar membro desse grupo.
8. (Opcional) Você pode escolher Criar grupo para definir um novo grupo. Esse recurso é útil se você quiser adicionar o usuário a um grupo com políticas de anexo diferentes das dos grupos existentes:

a. Na nova guia, em Nome do grupo de usuários, digite um nome para o novo grupo.

> **Note**
>
> O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS. Os nomes de grupos podem ser uma combinação de até 128 letras, dígitos e estes caracteres: adição (+), igual (=), vírgula (,), ponto (.), arroba (@) e hífen (-). Os nomes devem ser exclusivos dentro de uma conta. Eles não são diferenciados por letras maiúsculas e minúsculas. Por exemplo, você não pode criar dois grupos denominados TESTGROUP e testgroup.

b. Marque uma ou mais caixas de seleção para as políticas gerenciadas que você deseja anexar ao grupo. Você também pode criar uma nova política gerenciada escolhendo Criar política. Se fizer isso, volte para esta guia ou janela do navegador quando a nova política for concluída; escolha Atualizar e, em seguida, escolha a nova política para anexá-la ao grupo. Para obter mais informações, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.

c. Escolha Criar grupo de usuários.

d. Retorne à guia original e atualize a lista de grupos. Em seguida, marque a caixa de seleção do novo grupo.

9. Escolha Adicionar usuário aos grupos.

O console exibe uma mensagem de status informando que o usuário foi adicionado aos grupos que você especificou.

Para adicionar permissões copiando-as de outro usuário do IAM

Se você optar por adicionar permissões a um usuário do IAM por meio de cópia de permissões, o IAM copiará todas as associações de grupo, políticas gerenciadas anexadas, políticas em linha e quaisquer limites de permissões existentes do usuário especificado e as aplicará imediatamente ao usuário selecionado no momento.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome de usuário do IAM.
5. Na guia Permissões, escolha e selecione Adicionar permissões.
6. Na página Adicionar permissões, escolha Copiar permissões. A lista exibe os usuários do IAM disponíveis com sua associação aos grupos e às políticas anexadas.
7. Selecione o botão ao lado do usuário cujas permissões você deseja copiar.
8. Escolha Próximo para ver uma lista de alterações a serem feitas para o usuário. Em seguida, selecione Adicionar permissões.

O console exibe uma mensagem de status informando que as permissões foram copiadas do usuário do IAM que você especificou.

Para adicionar permissões anexando políticas diretamente ao usuário do IAM

Você pode anexar uma política gerenciada diretamente a um usuário do IAM. As permissões atualizadas são aplicadas imediatamente.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome de usuário do IAM.
5. Na guia Permissões, escolha e selecione Adicionar permissões.
6. Na página Adicionar permissões, escolha Anexar políticas diretamente. A lista Políticas de permissões exibe as políticas disponíveis juntamente com os tipos de política e entidades anexadas.

7. Selecione o botão de opção ao lado do nome da Política que você deseja anexar.
8. Escolha Próximo para ver uma lista de alterações a serem feitas para o usuário. Em seguida, selecione Adicionar permissões.

O console exibe uma mensagem de status informando que a política foi adicionada ao usuário do IAM especificado.

Para definir o limite de permissões de um usuário do IAM

Um limite de permissões é um atributo avançado para o gerenciamento de permissões na AWS que é usado para definir as permissões máximas que um usuário do IAM pode ter. Definir um limite de permissões restringe imediatamente as permissões do usuário do IAM ao limite, independentemente das outras permissões concedidas.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome do usuário do IAM cujo limite de permissões você deseja alterar.
5. Escolha a aba Permissões. Se necessário, abra a seção Limite de permissões e escolha Definir limite de permissões.
6. Na página Definir limite de permissões, em Políticas de permissões, selecione a política que você deseja usar para o limite de permissões.
7. Escolha Definir limite.

O console exibe uma mensagem de status informando que o limite de permissões foi adicionado.

Alteração das permissões de um usuário (console)

O IAM permite que você altere as permissões associadas a um usuário das seguintes maneiras:

- Editar uma política de permissões: edite a política em linha de um usuário, a política em linha do grupo do usuário ou edite uma política gerenciada associada ao usuário diretamente ou de um grupo. Se o usuário tiver um limite de permissões, você não poderá fornecer mais permissões além das permitidas pela política usada como o limite de permissões do usuário.
- Alterar o limite de permissões: altere a política usada como o limite de permissões para o usuário. Isso pode expandir ou restringir o número máximo de permissões que um usuário pode ter.

Editar uma política de permissões anexada a um usuário

A alteração das permissões atualiza o acesso do usuário imediatamente.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome do usuário do IAM cujo limite de permissões você deseja alterar.
5. Escolha a aba Permissões. Se necessário, abra a seção Limite de permissões.
6. Escolha o nome da política que deseja editar para visualizar os detalhes da política. Escolha a guia Entidades anexadas para visualizar outras entidades (usuários, grupos e perfis do IAM) que poderão ser afetadas caso edite a política.
7. Escolha a guia Permissões e revise as permissões concedidas pela política. Para fazer alterações nas permissões, escolha Editar.
8. Edite a política e resolva as recomendações de validação de política. Para obter mais informações, consulte Editar políticas do IAM.
9. Escolha Avançar, revise o resumo da política e, em seguida, selecione Salvar alterações.

   O console exibe uma mensagem de status informando que a política foi atualizada.

Para alterar o limite de permissões de um usuário

A alteração de um limite de permissões atualiza o acesso do usuário imediatamente.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome do usuário do IAM cujo limite de permissões você deseja alterar.
5. Escolha a aba Permissões. Se necessário, abra a seção Limite de permissões e, em seguida, escolha Alterar limite.
6. Selecione a política que você deseja usar para o limite de permissões.
7. Escolha Definir limite.

O console exibe uma mensagem de status informando que o limite de permissões foi alterado.

Para remover uma política de permissões de um usuário (console)

A remoção de uma política de permissões atualiza o acesso do usuário imediatamente.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Escolha o nome do usuário cujas políticas de permissões você deseja remover.
5. Escolha a aba Permissões.
6. Se quiser remover as permissões removendo uma política existente, visualize a coluna Anexado via para entender como o usuário está obtendo essa política antes de escolher Remover para remover a política:
   - Se a política for aplicável devido à associação ao grupo, escolher Remover removerá o usuário do grupo. Lembre-se de que você pode ter várias políticas anexadas a um único

grupo. Se você remover um usuário de um grupo, ele perderá o acesso a todas as políticas que recebeu por meio dessa associação ao grupo.

- Se for uma política gerenciada anexada diretamente ao usuário, escolher Remover desanexará a política do usuário. Isso não afetará a política em si nem qualquer outra entidade à qual política esteja anexada.
- Se a política for uma política incorporada em linha, a opção Remover removerá a política do IAM. As políticas em linha anexadas diretamente a um usuário existem somente para esse usuário.

Se a política tiver sido concedida ao usuário por meio de uma associação de grupo, o console exibirá uma mensagem de status informando que o usuário do IAM foi removido do grupo do IAM. Se a política estiver diretamente anexada ou em linha, a mensagem de status informará que a política foi removida.

Para remover o limite de permissões de um usuário (console)

A remoção do limite de permissões atualiza o acesso do usuário imediatamente.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Na lista Usuários, escolha o nome do usuário do IAM cujo limite de permissões você deseja remover.
5. Escolha a aba Permissões. Se necessário, abra a seção Limite de permissões.
6. Escolha Alterar limite. Para confirmar que você deseja remover o limite de permissões, na caixa de diálogo de confirmação, escolha Remover limite.

O console exibe uma mensagem de status informando que o limite de permissões foi removido.

Adição e remoção das permissões de um usuário (AWS CLI ou API da AWS)

Para adicionar ou remover permissões de forma programática, você deve adicionar ou remover as associações a grupos, anexar ou desanexar as políticas gerenciadas ou adicionar ou excluir as políticas em linha. Para obter mais informações, consulte os tópicos a seguir.

- Editar usuários em grupos do IAM
- Adicionar e remover permissões de identidade do IAM

Senhas de usuários na AWS

É possível gerenciar as senhas de usuários do IAM em sua conta. Usuários do IAM precisam de senhas para acessar o Console de gerenciamento da AWS. Os usuários não precisam de senhas para acessar os recursos da AWS de forma programática ao usar a AWS CLI, o Tools for Windows PowerShell, os SDKs ou APIs da AWS. Para esses ambientes, existe a opção de atribuir chaves de acesso aos usuários do IAM. Porém, existem alternativas mais seguras às chaves de acesso que recomendamos considerar primeiro. Para obter mais informações, consulte AWSCredenciais de segurança de.

> **Note**
>
> Se um de seus usuários do IAM perder ou esquecer a senha, você não poderá recuperá-la do IAM. Dependendo das suas configurações, o usuário ou o administrador devem criar uma nova senha.

Conteúdo

- Definir uma política de senhas de contas para usuários do IAM
- Gerenciar senhas para usuários do IAM
- Permitir que os usuários do IAM alterem as próprias senhas
- Como um usuário do IAM altera a própria senha

Definir uma política de senhas de contas para usuários do IAM

Você pode definir uma política de senha personalizada em sua Conta da AWS para especificar requisitos de complexidade e períodos de alternância obrigatórios para suas senhas de usuários

do IAM. Se você não definir uma política de senha personalizada, as senhas de usuário do IAM deverão atender à política de senha da AWS padrão. Para obter mais informações, consulte Opções de políticas de senha personalizadas.

Tópicos

- Regras para definir uma política de senha
- Permissões necessárias para definir uma política de senha
- Política de senha padrão
- Opções de políticas de senha personalizadas
- Para definir uma política de senha (console)
- Para alterar uma política de senha (console)
- Para excluir uma política de senhas personalizada (console)
- Definir uma política de senha (AWS CLI)
- Definir uma política de senha (API da AWS)

Regras para definir uma política de senha

A política de senha do IAM não se aplica à senha do Usuário raiz da conta da AWS ou às chaves de acesso do usuário do IAM. Se uma senha expirar, o usuário do IAM não poderá fazer login no Console de gerenciamento da AWS, mas poderá continuar a usar as chaves de acesso.

Quando você criar ou alterar uma política de senha, a maioria das configurações de política de senha será aplicada da próxima vez que seus usuários mudarem suas senhas. No entanto, algumas das configurações serão aplicadas imediatamente. Por exemplo:

- Quando os requisitos de tamanho mínimo e o tipo de caracteres forem alterados, as configurações serão aplicadas na próxima vez que os usuários mudarem suas senhas. Os usuários não serão forçados a mudar suas senhas existentes, mesmo se as senhas existentes não estiverem em conformidade com a política de senhas atualizada.
- Quando você definir um período de expiração de senha, ele será aplicado imediatamente. Por exemplo, suponhamos que você defina um período de expiração de senha de 90 dias. Nesse caso, a senha expira para todos os usuários do IAM cuja senha existente tenha sido configurada há mais de 90 dias. Esses usuários são obrigados a alterar sua senha na próxima vez que fizerem login.

Você não pode criar uma “política de bloqueio” para bloquear um usuário fora da conta após um número especificado de tentativas de login com falha. Para maior segurança, recomendamos que você combine uma política de senha forte com a autenticação multifator (MFA). Para obter mais informações sobre MFA, consulte AWSCódigo da autenticação multifator no IAM da.

Permissões necessárias para definir uma política de senha

Você deve configurar permissões para permitir que uma entidade (usuário ou função) do IAM visualize ou edite a política de senha da conta. Você pode incluir as seguintes ações de política de senha em uma política do IAM:

- `iam:GetAccountPasswordPolicy`: permite que a entidade visualize a política de senha para a conta
- `iam:DeleteAccountPasswordPolicy`: permite que a entidade exclua a política de senha personalizada para a conta e reverta para a política de senha padrão
- `iam:UpdateAccountPasswordPolicy`: permite que a entidade crie ou altere a política de senha personalizada para a conta

A política a seguir permite acesso total para visualizar e editar a política de senha da conta. Para saber mais sobre como criar uma política do IAM usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Para obter informações sobre as permissões necessárias para um usuário do IAM alterar sua própria senha, consulte Permitir que os usuários do IAM alterem as próprias senhas.

Política de senha padrão

Se um administrador não definir uma política de senha personalizada, as senhas de usuário do IAM deverão atender à política de senha padrão da AWS.

A política de senha padrão impõe as seguintes condições:

- Tamanho mínimo da senha é 8 caracteres e o máximo 128 caracteres.
- No mínimo três dos seguintes tipos de caracteres: maiúsculas, minúsculas, números e caracteres não alfanuméricos (`! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`)
- Não ser idêntico ao nome ou endereço de e-mail da sua Conta da AWS
- A senha nunca expira

Opções de políticas de senha personalizadas

Ao configurar uma política de senha personalizada para sua conta, você pode especificar as seguintes condições:

- Password minimum length (Comprimento mínimo da senha): especifique um mínimo de seis caracteres e um máximo de 128 caracteres.
- Força da senha: selecione qualquer uma das seguintes opções para definir a força das suas senhas de usuário do IAM:
  - Exigir pelo menos uma letra maiúscula do alfabeto latino (A–Z)
  - Exigir pelo menos uma letra minúscula do alfabeto latino (a–z)
  - Exigir pelo menos um número
  - Exigir pelo menos um caractere não alfanumérico `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Turn on password expiration (Ativar expiração de senha): você pode selecionar e especificar no mínimo um e no máximo 1.095 dias para que as senhas de usuário do IAM permaneçam válidas após serem definidas. Por exemplo, se você especificar uma validade de 90 dias, isso afetará imediatamente todos os seus usuários. Após a mudança, os usuários que têm senhas com mais de 90 dias precisam definir uma nova senha ao acessar o console. Usuários com senhas de 75 a 89 dias recebem um aviso do Console de gerenciamento da AWS sobre a validade da senha.

Os usuários do IAM podem alterar a senha a qualquer momento, desde que tenham recebido permissão para isso. Ao definir uma nova senha, o período de expiração para ela será reiniciado. Um usuário do IAM só pode ter uma senha válida por vez.

- Password expiration requires administrator reset (A expiração da senha requer redefinição do administrador): selecione esta opção para impedir que os usuários do IAM usem o Console de gerenciamento da AWS para atualizar suas próprias senhas depois que elas expirarem. Antes de selecionar esta opção, confirme se a sua Conta da AWS tem mais de um usuário com permissões administrativas para redefinir as senhas de usuário do IAM. Administradores com a permissão `iam:UpdateLoginProfile` podem redefinir senhas de usuário do IAM. Usuários do IAM com permissões `iam:ChangePassword` e chaves de acesso ativas podem redefinir sua própria senha do console do usuário do IAM programaticamente. Se você desmarcar esta caixa de seleção, os usuários do IAM com senhas expiradas ainda deverão definir uma nova senha para que possam acessar o Console de gerenciamento da AWS.
- Allow users to change their own password (Permitir que os usuários alterem suas próprias senhas): você pode permitir que todos os usuários do IAM na conta alterem suas próprias senhas. Isso dá aos usuários acesso à ação `iam:ChangePassword` somente para seu usuário e à ação `iam:GetAccountPasswordPolicy`. Essa opção não anexa uma política de permissões a cada usuário. Em vez disso, o IAM aplica as permissões no nível da conta a todos os usuários. Ou então, você pode permitir que apenas alguns usuários gerenciem suas próprias senhas. Para fazer isso, desmarque esta caixa de seleção. Para obter mais informações sobre o uso de políticas para limitar quem pode gerenciar senhas, consulte Permitir que os usuários do IAM alterem as próprias senhas.
- Prevent password reuse (Impedir a reutilização de senha): você pode impedir que os usuários do IAM reutilizem um número específico de senhas anteriores. Você pode especificar um número mínimo de 1 e um número máximo de 24 senhas anteriores que não podem ser repetidas.

Para definir uma política de senha (console)

Você pode usar o Console de gerenciamento da AWS para criar, alterar ou excluir uma política de senha. As alterações na política de senha se aplicam aos usuários do IAM criados após essa alteração de política e aos usuários existentes do IAM ao alterarem suas senhas.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.

2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione Configurações da conta.
4. Na seção Password policy (Política de senha), escolha Edit (Editar).
5. Escolha Custom (Personalizada) para usar uma política de senha personalizada.
6. Selecione as opções que você deseja aplicar à política de senha e escolha Save changes (Salvar alterações).
7. Confirme se deseja definir uma política de senha personalizada escolhendo Set custom (Definir personalizada).

O console exibe uma mensagem de status informando que os requisitos de senha para os usuários do IAM foram atualizados.

Para alterar uma política de senha (console)

Você pode usar o Console de gerenciamento da AWS para criar, alterar ou excluir uma política de senha. As alterações na política de senha se aplicam aos usuários do IAM criados após essa alteração de política e aos usuários existentes do IAM ao alterarem suas senhas.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione Configurações da conta.
4. Na seção Password policy (Política de senha), escolha Edit (Editar).
5. Selecione as opções que você deseja aplicar à política de senha e escolha Save changes (Salvar alterações).
6. Confirme se deseja definir uma política de senha personalizada escolhendo Set custom (Definir personalizada).

O console exibe uma mensagem de status informando que os requisitos de senha para os usuários do IAM foram atualizados.

Para excluir uma política de senhas personalizada (console)

Você pode usar o Console de gerenciamento da AWS para criar, alterar ou excluir uma política de senha. As alterações na política de senha se aplicam aos usuários do IAM criados após essa alteração de política e aos usuários existentes do IAM ao alterarem suas senhas.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, selecione Configurações da conta.
4. Na seção Password policy (Política de senha), escolha Edit (Editar).
5. Escolha IAM default (Padrão do IAM) para excluir a política de senha personalizada e escolha Save changes (Salvar alterações).
6. Confirme se deseja definir a política de senha padrão do IAM escolhendo Set default (Definir padrão).

   O console exibe uma mensagem de status informando que a política de senha está definida como padrão do IAM.

Definir uma política de senha (AWS CLI)

Você pode usar o AWS Command Line Interface para definir uma política de senha.

Para gerenciar a política de senha de conta personalizada a no AWS CLI

Execute os seguintes comandos:

- Para criar ou alterar a política de senha personalizada: `aws iam update-account-password-policy`
- Para exibir a política de senha: `aws iam get-account-password-policy`
- Para excluir a política de senha personalizada: `aws iam delete-account-password-policy`

Definir uma política de senha (API da AWS)

Você pode usar operações de AWS API para definir uma política de senha.

Para gerenciar a política de senha de conta personalizada na AWS API

Chame as seguintes operações:

- Para criar ou alterar a política de senha personalizada: `UpdateAccountPasswordPolicy`
- Para exibir a política de senha: `GetAccountPasswordPolicy`
- Para excluir a política de senha personalizada: `DeleteAccountPasswordPolicy`

Gerenciar senhas para usuários do IAM

Os usuários do IAM que usarem o Console de gerenciamento da AWS para trabalhar com recursos da AWS deverão ter uma senha para fazer login. Você pode criar, alterar ou excluir uma senha de um usuário do IAM em sua conta da AWS.

Depois de ter atribuído uma senha a um usuário, o usuário pode fazer login no Console de gerenciamento da AWS usando o URL de login para a sua conta, que é semelhante a:

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Para obter mais informações sobre como usuários do IAM fazem login no Console de gerenciamento da AWS, consulte Como fazer login na conta da AWS no Guia do usuário do Início de Sessão da AWS.

Mesmo que os usuários tenham suas próprias senhas, eles ainda precisarão de permissões para acessar seus recursos da AWS. Por padrão, um usuário não tem nenhuma permissão. Para conceder aos seus usuários as permissões de que precisam, atribua políticas a eles ou aos grupos aos quais pertencem. Para obter informações sobre a criação de usuários e grupos, consulte Identidades do IAM. Para obter informações sobre o uso de políticas para definir permissões, consulte Alterar permissões de um usuário do IAM.

É possível conceder aos usuários permissão para alterar as próprias senhas. Para obter mais informações, consulte Permitir que os usuários do IAM alterem as próprias senhas. Para obter informações sobre como os usuários acessam a página de login, consulte Como fazer login na AWS, no Guia do usuário do Início de Sessão da AWS.

Tópicos

- Criar, alterar ou excluir uma senha de usuário do IAM (console)

Criar, alterar ou excluir uma senha de usuário do IAM (console)

Você pode usar o Console de gerenciamento da AWS para gerenciar senhas de seus usuários do IAM.

As necessidades de acesso dos usuários podem mudar com o tempo. Talvez seja necessário habilitar um usuário destinado ao acesso à CLI para ter acesso ao console, alterar a senha de um usuário porque ele recebeu o e-mail com suas credenciais ou excluir um usuário quando ele sair da organização ou não precisar mais de acesso à AWS.

Para criar uma senha de usuário do IAM (console)

Use este procedimento para conceder acesso ao console a um usuário, criando uma senha associada ao nome de usuário.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Escolha o nome do usuário cuja senha você deseja criar.
5. Escolha a guia Credenciais de segurança e, em Login no console, escolha Habilitar acesso ao console.
6. Na caixa de diálogo Habilitar acesso ao console, selecione Redefinir senha e escolha se o IAM deve gerar uma senha ou criar uma senha personalizada:
   - Para que o IAM gere uma senha, escolha a opção Autogenerated password (Senha gerada automaticamente).
   - Para criar uma senha personalizada, escolha Senha personalizada e digite a senha.

   > **Note**
   >
   > A senha que você criou deve cumprir a política de senhas da conta.

7. Para exigir que o usuário crie uma nova senha ao fazer login, escolha Exigir alteração de senha no próximo login.
8. Para exigir que o usuário use a nova senha imediatamente, selecione Revogar sessões ativas do console. Esse procedimento anexa uma política em linha ao usuário do IAM que nega o acesso do usuário aos recursos caso as credenciais sejam mais antigas do que o período especificado pela política.
9. Escolha Redefinir senha
10. A caixa de diálogo Senha do console informa que você ativou a nova senha do usuário. Para visualizar a senha e compartilhá-la com o usuário, escolha Mostrar na caixa de diálogo Senha do console. Selecione Baixar arquivo .csv para baixar um arquivo com as credenciais do usuário.

> **Important**
>
> Por motivos de segurança, você não pode acessar a senha depois de concluir esta etapa, mas pode criar uma nova senha a qualquer momento.

O console exibe uma mensagem de status informando que o acesso ao console foi habilitado.

Para alterar a senha para um usuário do IAM (console)

Use este procedimento para atualizar uma senha associada ao nome de usuário.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Escolha o nome do usuário cuja senha você deseja alterar.
5. Escolha a guia Credenciais de segurança e, em Login no console, escolha Gerenciar acesso ao console.
6. Na caixa de diálogo Gerenciar acesso ao console, selecione Redefinir senha e, em seguida, escolha se deseja que o IAM gere uma senha ou crie uma senha personalizada:

- Para que o IAM gere uma senha, escolha a opção Autogenerated password (Senha gerada automaticamente).
- Para criar uma senha personalizada, escolha Senha personalizada e digite a senha.

> **Note**
>
> A senha que você criou deve cumprir a política de senhas da conta.

7. Para exigir que o usuário crie uma nova senha ao fazer login, escolha Exigir alteração de senha no próximo login.
8. Para exigir que o usuário use a nova senha imediatamente, selecione Revogar sessões ativas do console. Esse procedimento anexa uma política em linha ao usuário do IAM que nega o acesso do usuário aos recursos caso as credenciais sejam mais antigas do que o período especificado pela política.
9. Escolha Redefinir senha
10. A caixa de diálogo Senha do console informa que você ativou a nova senha do usuário. Para visualizar a senha e compartilhá-la com o usuário, escolha Mostrar na caixa de diálogo Senha do console. Selecione Baixar arquivo .csv para baixar um arquivo com as credenciais do usuário.

> **Important**
>
> Por motivos de segurança, você não pode acessar a senha depois de concluir esta etapa, mas pode criar uma nova senha a qualquer momento.

O console exibe uma mensagem de status informando que o acesso ao console foi atualizado.

Para excluir (desabilitar) a senha de um usuário do IAM (console)

Use este procedimento para excluir uma senha associada ao nome de usuário, removendo o acesso do usuário ao console.

> **Important**
>
> Você pode impedir que um usuário do IAM acesse o Console de gerenciamento da AWS removendo a senha dele. Isso impede que ele faça login no Console de gerenciamento da AWS usando suas credenciais de login. Isso não altera as permissões do usuário nem o impede de acessar o console usando uma função assumida. Se o usuário tiver chaves de acesso ativas, elas continuarão funcionando e permitirão o acesso por meio da AWS CLI, do Tools for Windows PowerShell, da API da AWS ou do AWS Console Mobile Application.

Console

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como fazer login na AWS no Guia do usuário do AWS Sign-In.
2. Na Página inicial do console do IAM, no painel de navegação à esquerda, insira sua consulta na caixa de texto Pesquisar IAM.
3. No painel de navegação, escolha Users.
4. Escolha o nome do usuário cuja senha você deseja excluir.
5. Escolha a guia Credenciais de segurança e, em Login no console, escolha Gerenciar acesso ao console.
6. Para requerer que o usuário pare de usar o console imediatamente, selecione Revogar sessões ativas do console. Esse procedimento anexa uma política em linha ao usuário do IAM que nega o acesso do usuário aos recursos caso as credenciais sejam mais antigas do que o período especificado pela política.
7. Escolha Desabilitar acesso

   O console exibe uma mensagem de status informando que o acesso ao console foi desabilitado.

Criar, alterar ou excluir uma senha de usuário do IAM (AWS CLI)

Você pode usar o AWS CLI para gerenciar senhas de seus usuários do IAM.

Para criar uma senha (AWS CLI)

1. (Opcional) Para determinar se um usuário tem uma senha, execute este comando: aws iam get-login-profile

2. Para criar uma senha, execute o seguinte comando: aws iam create-login-profile

Para alterar a senha de um usuário (AWS CLI)

1. (Opcional) Para determinar se um usuário tem uma senha, execute este comando: aws iam get-login-profile
2. Para alterar uma senha, execute o seguinte comando: aws iam update-login-profile

Para excluir (desabilitar) a senha de um usuário (AWS CLI)

1. (Opcional) Para determinar se um usuário tem uma senha, execute este comando: aws iam get-login-profile
2. (Opcional) Para determinar quando uma senha foi usada pela última vez, execute este comando: aws iam get-login-user
3. Para excluir uma senha, execute o seguinte comando: aws iam delete-login-profile

> **Important**
>
> Quando você excluir a senha de um usuário, o usuário não poderá mais fazer login no Console de gerenciamento da AWS. Se o usuário tiver chaves de acesso ativas, elas continuarão funcionando e permitirão o acesso por meio da AWS CLI, do Tools for Windows PowerShell ou de chamadas de função da API da AWS. Ao usar a AWS CLI, o Tools for Windows PowerShell ou a API da AWS para excluir um usuário da sua Conta da AWS, você deve primeiro excluir a senha usando essa operação. Para obter mais informações, consulte Exclusão de um usuário do IAM (AWS CLI).

Para revogar as sessões ativas do console de um usuário antes de um horário especificado (AWS CLI)

1. Para incorporar uma política em linha que revogue as sessões ativas do console de um usuário do IAM antes de um horário especificado, use a seguinte política em linha e execute este comando: aws iam put-user-policy

   Essa política em linha nega todas as permissões e inclui a chave de condição `aws:TokenIssueTime`. Ela revoga as sessões ativas do console do usuário antes do horário

especificado no elemento `Condition` da política em linha. Substitua o valor da chave de condição `aws:TokenIssueTime` pelos seus próprios valores.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (Opcional) Para listar os nomes das políticas em linha incorporadas no usuário do IAM, execute este comando: aws iam list-user-policies
3. (Opcional) Para listar os nomes das políticas em linha incorporadas no usuário do IAM, execute este comando: aws iam list-user-policies

Criar, alterar ou excluir uma senha de usuário do IAM (API da AWS)

Você pode usar a API da AWS para gerenciar as senhas de seus usuários do IAM.

Para criar uma senha (API da AWS)

1. (Opcional) Para determinar se um usuário tem uma senha, chame esta operação: GetLoginProfile
2. Para criar uma senha, chame esta operação: CreateLoginProfile

Para alterar a senha de um usuário (API da AWS)

1. (Opcional) Para determinar se um usuário tem uma senha, chame esta operação: GetLoginProfile
2. Para alterar uma senha, chame esta operação: UpdateLoginProfile

Para excluir (desabilitar) a senha de um usuário (API da AWS)

1. (Opcional) Para determinar se um usuário tem uma senha, execute este comando: GetLoginProfile
2. (Opcional) Para determinar quando uma senha foi usada pela última vez, execute este comando: GetUser
3. Para excluir uma senha, execute o seguinte comando: DeleteLoginProfile

> **Important**
>
> Quando você excluir a senha de um usuário, o usuário não poderá mais fazer login no Console de gerenciamento da AWS. Se o usuário tiver chaves de acesso ativas, elas continuarão funcionando e permitirão o acesso por meio da AWS CLI, do Tools for Windows PowerShell ou de chamadas de função da API da AWS. Ao usar a AWS CLI, o Tools for Windows PowerShell ou a API da AWS para excluir um usuário da sua Conta da AWS, você deve primeiro excluir a senha usando essa operação. Para obter mais informações, consulte Exclusão de um usuário do IAM (AWS CLI).

Para revogar as sessões ativas do console de um usuário antes de um horário especificado (API AWS)

1. Para incorporar uma política em linha que revogue as sessões ativas do console de um usuário do IAM antes de um horário especificado, use a seguinte política em linha e execute este comando: PutUserPolicy

   Essa política em linha nega todas as permissões e inclui a chave de condição `aws:TokenIssueTime`. Ela revoga as sessões ativas do console do usuário antes do horário especificado no elemento `Condition` da política em linha. Substitua o valor da chave de condição `aws:TokenIssueTime` pelos seus próprios valores.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
```

```
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (Opcional) Para listar os nomes das políticas em linha incorporadas no usuário do IAM, execute este comando: ListUserPolicies
3. (Opcional) Para listar os nomes das políticas em linha incorporadas no usuário do IAM, execute este comando: GetUserPolicy

Permitir que os usuários do IAM alterem as próprias senhas

> **Note**
>
> Usuários com identidades federadas usarão o processo definido pelo provedor de identidade para alterar as senhas. Como prática recomendada, exija que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias.

Você pode conceder aos usuários do IAM permissão para alterar as próprias senhas para fazer login no Console de gerenciamento da AWS. É possível fazer isso de duas formas:

- Permitir que todos os usuários do IAM na conta alterem as próprias senhas.
- Permitir que apenas usuários selecionados do IAM alterem as próprias senhas. Nesse cenário, você desabilita a opção para todos os usuários alterarem suas próprias senhas e usa uma política do IAM para conceder permissões a apenas alguns usuários. Essa abordagem permite que esses usuários mudem suas próprias senhas e, opcionalmente, outras credenciais, como suas próprias chaves de acesso.

> **Important**
>
> Recomendamos que você defina uma política de senha personalizada que exija que os usuários do IAM criem senhas fortes.

Para permitir que todos os usuários do IAM alterem as próprias senhas

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Configurações da conta.
3. Na seção Password policy (Política de senha), escolha Edit (Editar).
4. Escolha Custom (Personalizada) para usar uma política de senha personalizada.
5. Selecione Allow users to change their own password (Permitir que os usuários alterem sua própria senha) e clique em Save changes (Salvar alterações). Isso permite que todos os usuários na conta acessem a ação `iam:ChangePassword` somente para seu usuário e para a ação `iam:GetAccountPasswordPolicy`.
6. Forneça aos usuários as seguintes instruções para alterar as senhas: Como um usuário do IAM altera a própria senha.

AWS CLI

Execute este comando:

- `aws iam update-account-password-policy`

API

Para atualizar a política de senha da conta, chame a seguinte operação:

- `UpdateAccountPasswordPolicy`

Para permitir que usuários selecionados do IAM alterem as próprias senhas

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Configurações da conta.
3. Na seção Password policy (Políticas da conta), verifique se a opção Allow users to change their own password (Permitir que os usuários alterem a própria senha) não está selecionada. Se essa caixa de seleção estiver marcada, todos os usuários poderão alterar as próprias senhas. (Consulte o procedimento anterior).
4. Crie usuários capazes de alterar as próprias senhas, se eles ainda não existirem. Para obter detalhes, consulte Criar um usuário do IAM na Conta da AWS.
5. (Opcional) Crie um grupo do IAM para usuários que devem ter permissão para alterar as senhas e adicione os usuários da etapa anterior ao grupo. Para obter detalhes, consulte Grupos de usuários do IAM.
6. Atribua a política a seguir ao grupo. Para obter mais informações, consulte Gerenciar políticas do IAM.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Essa política concede acesso à ação ChangePassword, que permite aos usuários alterar somente as próprias senhas do console, da AWS CLI, do Tools for Windows PowerShell ou ou da API. Isso também concede acesso à ação GetAccountPasswordPolicy, que permite ao usuário ver a política de senha atual; essa permissão é necessária para que o usuário possa exibir a política de senhas da conta na página Change Password (Alterar senha). O usuário deve poder ler a política de senha atual para certificar-se de que a senha alterada esteja de acordo com os requisitos da política.

7. Forneça aos usuários as seguintes instruções para alterar as senhas: Como um usuário do IAM altera a própria senha.

Para obter mais informações

Para obter mais informações sobre o gerenciamento de credenciais, consulte os seguintes tópicos:

- Permitir que os usuários do IAM alterem as próprias senhas
- Senhas de usuários na AWS
- Definir uma política de senhas de contas para usuários do IAM
- Gerenciar políticas do IAM
- Como um usuário do IAM altera a própria senha

Como um usuário do IAM altera a própria senha

Se você tiver recebido permissão para alterar a própria senha de usuário do IAM, poderá usar uma página especial no Console de gerenciamento da AWS para fazer isso. Você também pode usar a AWS CLI ou a API da AWS.

Tópicos

- Permissões obrigatórias
- Como os usuários do IAM alteram a própria senha (console)
- Como os usuários do IAM alteram suas próprias senhas (AWS CLI ou API da AWS)

Permissões obrigatórias

Para mudar a senha do seu próprio usuário do IAM, você deve ter as permissões da seguinte política: AWS: permite que os usuários do IAM alterem suas próprias senhas do console na página Credenciais de segurança.

Como os usuários do IAM alteram a própria senha (console)

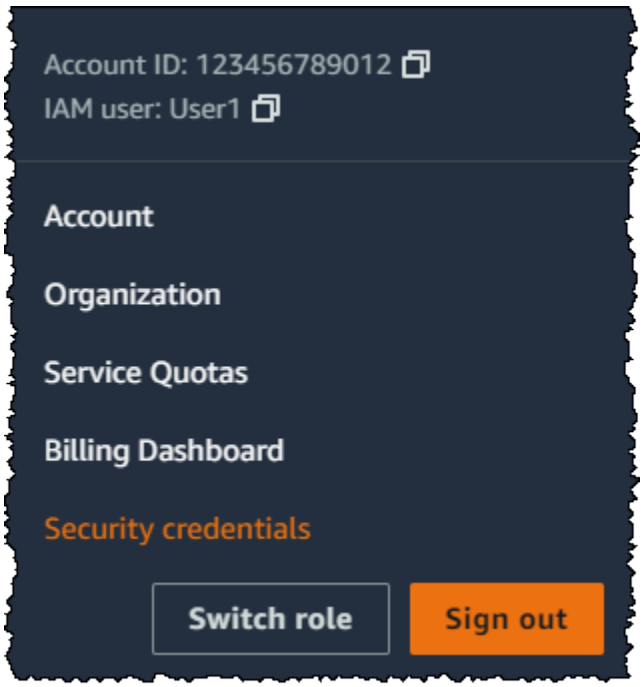
O procedimento a seguir descreve como os usuários do IAM podem usar o Console de gerenciamento da AWS para alterar sua própria senha.

Para alterar sua própria senha de usuário do IAM (console)

1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

   > **Note**
   >
   > Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

   Para obter o ID da Conta da AWS, fale com o administrador.
2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na guia Credenciais do AWS IAM, escolha Atualizar senha.
4. Em Current password (Senha atual), insira a sua senha atual. Insira uma nova senha em New password (Nova senha) e Confirm new password (Confirmar nova senha). Em seguida, selecione Atualizar senha.

> **Note**
>
> Se a conta tiver uma política de senha, a nova senha deverá atender aos requisitos dessa política. Para obter mais informações, consulte Definir uma política de senhas de contas para usuários do IAM.

Como os usuários do IAM alteram suas próprias senhas (AWS CLI ou API da AWS)

O procedimento a seguir descreve como os usuários do IAM podem usar a AWS CLI ou a API da AWS para alterar sua própria senha.

Para alterar sua própria senha do IAM, use o seguinte:

- AWS CLI: `aws iam change-password`
- AWS API da: `ChangePassword`

Gerenciar chaves de acesso para usuários do IAM

> **Important**
>
> Como prática recomendada, use credenciais de segurança temporárias (como perfis do IAM), em vez de criar credenciais de longo prazo, como as chaves de acesso. Antes de criar chaves de acesso, avalie as alternativas às chaves de acesso de longo prazo.

As chaves de acesso são credenciais de longo prazo para um usuário do IAM ou o Usuário raiz da conta da AWS. Você pode usar chaves de acesso para assinar solicitações programáticas na AWS CLI ou na API da AWS (diretamente ou usando o SDK da AWS). Para obter mais informações, consulte Acesso programático com credenciais de segurança da AWS.

As chaves de acesso consistem em duas partes: um ID de chave de acesso (por exemplo, `AKIAIOSFODNN7EXAMPLE`) e uma chave de acesso secreta (por exemplo, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Você deve usar o ID da chave de acesso e a chave de acesso secreta em conjunto para autenticar suas solicitações.

Ao criar um par de chaves de acesso, salve o ID de chave de acesso e a chave de acesso secreta em um local seguro. A chave de acesso secreta pode ser recuperada somente no momento em que você a cria. Se tiver perdido a chave de acesso secreta, você deverá excluir a chave de acesso e criar uma nova. Para obter mais instruções, consulte Atualizar chaves de acesso.

É possível ter um máximo de duas chaves de acesso por usuário.

> **Important**
>
> Usuários do IAM com chaves de acesso são um risco à segurança da conta. Gerencie suas chaves de acesso com segurança. Não as forneça a terceiros não autorizados, mesmo que seja para ajudar a localizar os identificadores da sua conta. Ao fazer isso, você pode dar a alguém acesso permanente à sua conta.
> Ao trabalhar com chaves de acesso, tenha em mente o seguinte:
>
> - NÃO use as credenciais de usuário-raiz da sua conta para criar chaves de acesso.
> - NÃO coloque chaves de acesso literais ou informações de credenciais em seus arquivos de aplicações.

- NÃO inclua arquivos que contenham informações de chaves de acesso ou de credenciais em sua área de projeto.
- As informações de chaves de acesso ou de credenciais armazenadas no arquivo de credenciais da AWS compartilhado são armazenadas em texto simples.

Recomendações de monitoramento

Após criar chaves de acesso:

- Use o AWS CloudTrail para monitorar o uso da chave de acesso e detectar quaisquer tentativas de acesso não autorizadas. Para obter mais informações, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.
- Configure alarmes do CloudWatch para notificar os administradores sobre tentativas de acesso negado para ajudar a detectar atividades maliciosas. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch.
- Revise, atualize e exclua regularmente as chaves de acesso conforme necessário.

Os tópicos a seguir detalham as tarefas de gerenciamento associadas a chaves de acesso.

Tópicos

- Controle o uso de chaves de acesso anexando uma política em linha a um usuário do IAM
- Permissões necessárias para gerenciar chaves de acesso
- Como os usuários do IAM podem gerenciar suas próprias chaves de acesso
- Como um administrador do IAM pode gerenciar as chaves de acesso do usuário do IAM
- Atualizar chaves de acesso
- Proteger chaves de acesso

Controle o uso de chaves de acesso anexando uma política em linha a um usuário do IAM

Outra prática recomendada é exigir que as workloads usem credenciais temporárias com perfis do IAM para acessar a AWS. Os usuários do IAM com chaves de acesso devem ter acesso com privilégios mínimos e ter a autenticação multifator (MFA) ativada. Para obter mais informações sobre como presumir um perfil do IAM, consulte Métodos para assumir um perfil.

No entanto, se você estiver criando um teste de prova de conceito de uma automação de serviço ou outro caso de uso de curto prazo e optar por executar workloads usando um usuário do IAM com chaves de acesso, recomendamos usar condições da política para restringir ainda mais o acesso das credenciais de usuário do IAM.

Nessa situação, você pode criar uma política com limite de tempo que expira as credenciais após o tempo especificado ou, se estiver executando uma workload em uma rede segura, você pode usar uma política de restrição de IP.

Para ambos os casos de uso, você pode usar uma política em linha anexada ao usuário do IAM que tem chaves de acesso.

Para configurar uma política com limite de tempo para um usuário do IAM

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Usuários e escolha o usuário para o caso de uso de curto prazo. Se você ainda não criou o usuário, pode criar o usuário agora.
3. Na página Detalhes do usuário, selecione a guia Permissões.
4. Escolha Adicionar permissões e, em seguida, selecione Criar política em linha.
5. Na seção Editor de políticas, selecione JSON para exibir o editor JSON.
6. No editor JSON, insira a política a seguir, substituindo o valor do `aws:CurrentTime`carimbo de data/hora pela data e hora de expiração desejadas:

   JSON

```
{
"Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
      "DateGreaterThan": {
      "aws:CurrentTime": "2025-03-01T00:12:00Z"
        }
      }
    }
```

```
  ]
}
```

Essa política usa o efeito `Deny` para restringir todas as ações em todos os recursos após a data especificada. A condição `DateGreaterThan` compara a hora atual com o carimbo de data/hora que você definiu.

7. Selecione Avançar para ir para a página Revisar e criar. Em detalhes da Política, em Nome da política, insira um nome para a política e escolha Criar política.

Depois que a política é criada, ela é exibida na guia Permissões do usuário. Quando a hora atual for maior ou igual à hora especificada na política, o usuário não terá mais acesso aos recursos da AWS. Certifique-se de informar os desenvolvedores da workload sobre a data de expiração que você especificou para essas chaves de acesso.

Para configurar uma política de restrição de IP para um usuário do IAM

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários e selecione o usuário que executará a workload na rede segura. Se você ainda não criou o usuário, pode criar o usuário agora.
3. Na página Detalhes do usuário, selecione a guia Permissões.
4. Escolha Adicionar permissões e, em seguida, selecione Criar política em linha.
5. Na seção Editor de políticas, selecione JSON para exibir o editor JSON.
6. Copie a seguinte política do IAM para o editor JSON e altere os endereços IPv4 ou IPv6 públicos ou os intervalos de acordo com suas necessidades. Você pode usar https://checkip.amazonaws.com para determinar seu endereço IP público. Você pode especificar endereços IP individuais ou intervalos de endereços IP usando a notação de barra. Para obter mais informações, consulte aws:SourceIp.

> **Note**
>
> Os endereços IP não devem ser ofuscados por uma VPN ou um servidor proxy.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"IpRestrictionIAMPolicyForIAMUser",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Este exemplo de política nega o uso das chaves de acesso de um usuário do IAM com essa política aplicada, a menos que a solicitação tenha sido originada das redes (especificadas na notação CIDR) "203.0.113.0/24", "2001:DB8:1234:5678::/64" ou do endereço IP específico "203.0.114.1"

7. Selecione Avançar para ir para a página Revisar e criar. Em detalhes da Política, em Nome da política, insira um nome para a política e escolha Criar política.

Depois que a política é criada, ela é exibida na guia Permissões do usuário.

Você também pode aplicar essa política como uma política de controle de serviço (SCP) em várias contas da AWS bo AWS Organizations. Recomendamos usar uma condição adicional,

`aws:PrincipalArn`, para que essa declaração de política se aplique apenas aos usuários do IAM nas contas da AWS sujeitas a esse SCP. A política a seguir inclui essa atualização:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IpRestrictionServiceControlPolicyForIAMUsers",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "NotIpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64",
            "203.0.114.1"
          ]
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::*:user/*"
        }
      }
    }
  ]
}
```

Permissões necessárias para gerenciar chaves de acesso

> **Note**
>
> `iam:TagUser` é uma permissão opcional para adicionar e editar descrições da chave de acesso. Para obter mais informações, consulte . Marcar usuários do IAM

Para criar chaves de acesso para seu próprio usuário do IAM, você deve ter as permissões na seguinte política:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Para atualizar chaves de acesso para seu próprio usuário do IAM, é necessário ter as permissões da política a seguir:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
```

```
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Como os usuários do IAM podem gerenciar suas próprias chaves de acesso

Os administradores do IAM podem conceder aos usuários do IAM a permissão para autogerenciar as chaves de acesso anexando a política descrita em Permissões necessárias para gerenciar chaves de acesso.

Com essas permissões, o usuário do IAM pode usar os procedimentos a seguir para criar, ativar, desativar e excluir as chaves de acesso associadas ao nome de usuário.

Tópicos

- Criar uma chave de acesso para você mesmo (console)
- Desativar a chave de acesso (console)
- Ativar a chave de acesso (console)
- Excluir a chave de acesso (console)

Criar uma chave de acesso para você mesmo (console)

Caso tenha recebido as permissões apropriadas, você poderá usar o Console de gerenciamento da AWS para criar chaves de acesso para você mesmo.

Para criar, modificar ou excluir suas próprias chaves de acesso (console)

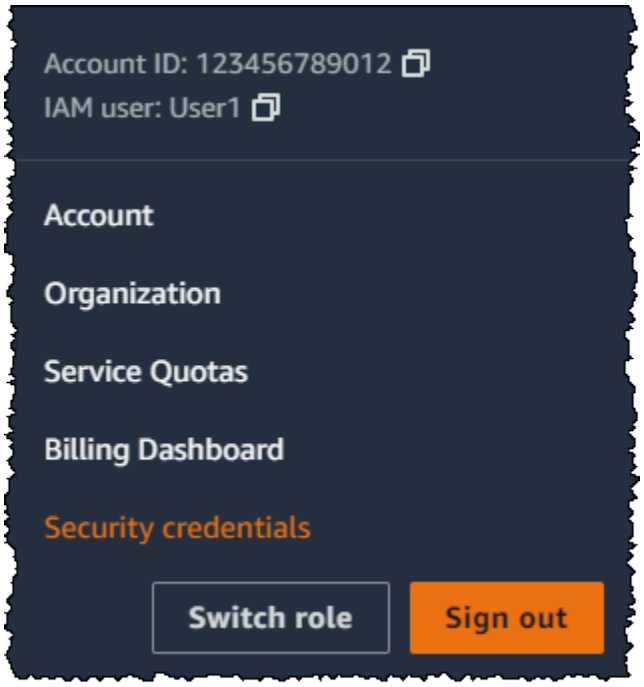
1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

> **Note**
>
> Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login

principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

Para obter o ID da Conta da AWS, fale com o administrador.

2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na seção Chaves de acesso, escolha Criar chave de acesso. Caso você já tenha duas chaves de acesso, este botão ficará desativado, e você deverá excluir uma chave de acesso antes de criar uma nova.
4. Na página Access key best practices and alternatives (Práticas recomendadas e alternativas da chave de acesso), escolha seu caso de uso para saber mais sobre outras opções que ajudam a evitar a criação de uma chave de acesso de longo prazo. Se você determinar que seu caso de uso ainda requer uma chave de acesso, escolha Other (Outro) e escolha Next (Avançar).
5. (Opcional) Defina um valor de etiqueta de descrição para a chave de acesso. Essa ação adiciona um par de chave-valor de etiqueta ao usuário do IAM. Isso pode ajudar a identificar e a atualizar as chaves de acesso posteriormente. A chave da etiqueta é definida como o ID da chave de acesso. O valor da etiqueta é definido com a descrição da chave de acesso que você especifica. Quando terminar, escolha Create access key (Criar chave de acesso).
6. Na página Retrieve access keys (Recuperar chaves de acesso), escolha Show (Exibir) para revelar o valor da chave de acesso secreta do usuário ou Download .csv file (Baixar o

arquivo.csv). Essa é a única oportunidade de salvar a chave de acesso secreta. Depois de salvar a chave de acesso secreta em um local seguro, escolha Done (Concluído).

Desativar a chave de acesso (console)

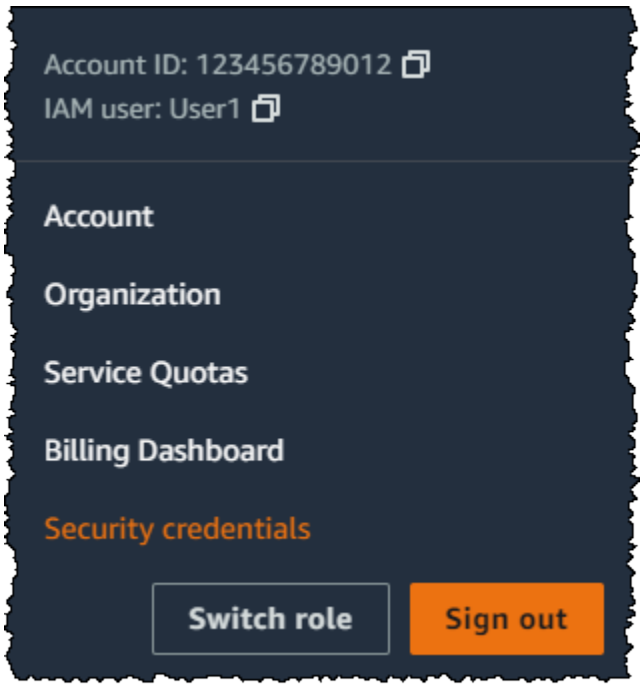
Caso tenha recebido as permissões apropriadas, você poderá usar o Console de gerenciamento da AWS para desativar a chave de acesso.

Para desativar uma chave de acesso

1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

   > **Note**
   >
   > Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

   Para obter o ID da Conta da AWS, fale com o administrador.
2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na seção Access keys (Chaves de acesso), localize a chave que você deseja desativar, escolha Actions (Ações) e escolha Deactivate (Desativar). Quando a confirmação for solicitada, escolha Deactivate (Desativar). A chave de acesso desativada ainda conta para o limite de duas chaves de acesso.

Ativar a chave de acesso (console)

Caso tenha recebido as permissões apropriadas, você poderá usar o Console de gerenciamento da AWS para ativar a chave de acesso.

Para ativar uma chave de acesso

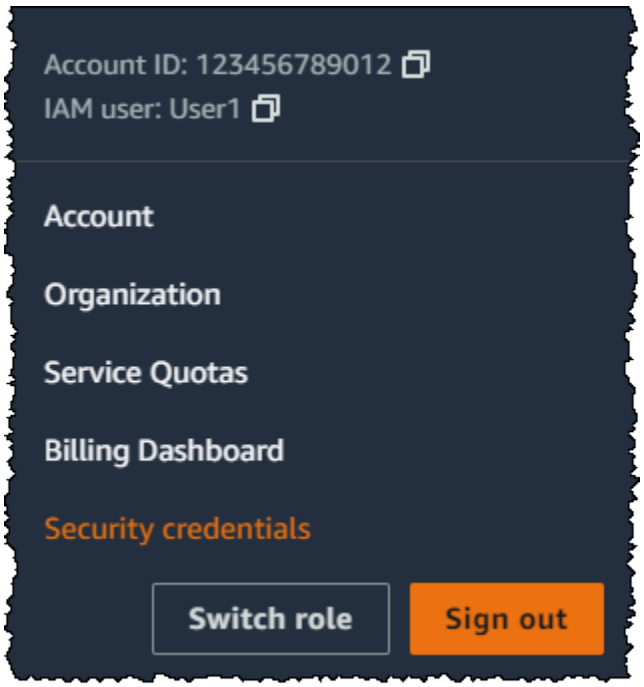
1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

> **Note**
>
> Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login

principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

Para obter o ID da Conta da AWS, fale com o administrador.

2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na seção Access keys (Chaves de acesso), localize a chave que deseja ativar, escolha Actions (Ações) e escolha Activate (Ativar).

Excluir a chave de acesso (console)

Caso tenha recebido as permissões apropriadas, você poderá usar o Console de gerenciamento da AWS para excluir a chave de acesso.

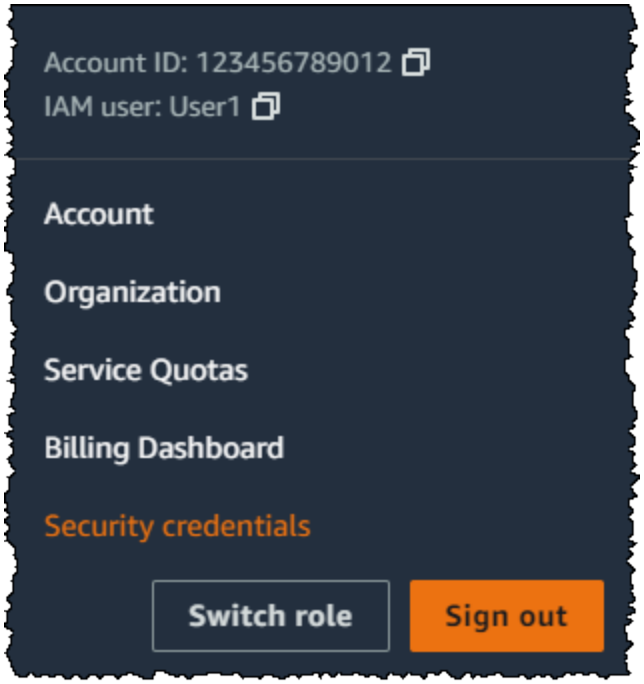
Para excluir uma chave de acesso quando não precisar mais dela

1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

> **Note**
>
> Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

Para obter o ID da Conta da AWS, fale com o administrador.

2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na seção Access keys (Chaves de acesso), localize a chave que deseja excluir, escolha Actions (Ações) e escolha Delete (Excluir). Siga as instruções na caixa de diálogo para primeiro Deactivate (Desativar) e depois confirme a exclusão. Recomendamos verificar se a chave de acesso não está mais em uso antes de excluí-la permanentemente.

Como um administrador do IAM pode gerenciar as chaves de acesso do usuário do IAM

Os administradores do IAM podem criar, ativar, desativar e excluir as chaves de acesso associadas a usuários do IAM individuais. Eles também podem listar os usuários do IAM na conta que têm chaves de acesso e localizar qual usuário do IAM tem uma chave de acesso específica.

Tópicos

- Como criar uma chave de acesso para um usuário do IAM
- Para desativar uma chave de acesso para um usuário do IAM
- Para ativar uma chave de acesso para um usuário do IAM
- Como excluir uma chave de acesso para um usuário do IAM
- Para listar as chaves de acesso de um usuário do IAM
- Para exibir todos os IDs de chave de acesso dos usuários na conta
- Para usar um ID de chave de acesso para localizar um usuário
- Para localizar o uso mais recente de um ID de chave de acesso

Como criar uma chave de acesso para um usuário do IAM

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Escolha o nome do usuário para ir para a página de detalhes do usuário.
4. Na guia Credenciais de segurança, na seção Chaves de acesso, escolha Criar chave de acesso.

   Se esse botão estiver desativado, você deverá excluir uma das chaves de acesso existentes antes de criar outra.
5. Na página Access key best practices and alternatives (Práticas recomendadas e alternativas de chaves de acesso), analise as práticas recomendadas e alternativas. Escolha seu caso de uso para saber mais sobre outras opções que ajudam a evitar a criação de uma chave de acesso de longo prazo.

6. Se você determinar que seu caso de uso ainda requer uma chave de acesso, escolha Other (Outro) e escolha Next (Avançar).
7. (Opcional) Na página Definir tag de descrição, você pode adicionar uma tag de descrição à chave de acesso para ajudar a rastreá-la. Selecione Criar chave de acesso.
8. Na página Retrieve access keys (Recuperar chaves de acesso), escolha Show (Exibir) para revelar o valor da chave de acesso secreta do usuário.
9. Para salvar o ID da chave de acesso e a chave de acesso secreta em um arquivo `.csv` em um local seguro no computador, escolha o botão Download .csv file (Baixar arquivo .csv).

> **Important**
>
> Esta é a única oportunidade de visualizar ou fazer download da chave de acesso recém-criada, e você não poderá recuperá-la posteriormente. Certifique-se de manter a chave de acesso protegida.

Quando você cria uma chave de acesso para seu usuário, o par de chaves é ativo por padrão, e você pode usar o par imediatamente.

AWS CLI

Execute este comando:

- `aws iam create-access-key`

API

Chame a seguinte operação:

- `CreateAccessKey`

Para desativar uma chave de acesso para um usuário do IAM

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.

3. Escolha o nome do usuário para ir para a página de detalhes do usuário.
4. Na guia Credenciais de segurança, na seção Chaves de acesso, escolha o menu suspenso Ações e, em seguida, Desativar.
5. Na caixa de diálogo Desativar, confirme que você deseja desativar a chave de acesso selecionando Desativar

Depois que uma chave de acesso é desativada, ela não pode mais ser usada por chamadas de API. Você poderá ativá-la novamente, se necessário.

AWS CLI

Execute este comando:

- `aws iam update-access-key`

API

Chame a seguinte operação:

- `UpdateAccessKey`

Para ativar uma chave de acesso para um usuário do IAM

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Escolha o nome do usuário para ir para a página de detalhes do usuário.
4. Na guia Credenciais de segurança, na seção Chaves de acesso, escolha o menu suspenso Ações e, em seguida, Ativar.

Depois que uma chave de acesso é ativada, ela pode ser usada por chamadas de API. Você poderá desativá-la novamente, se necessário.

AWS CLI

Execute este comando:

- `aws iam update-access-key`

API

Chame a seguinte operação:

- `UpdateAccessKey`

Como excluir uma chave de acesso para um usuário do IAM

Depois que uma chave de acesso tiver sido desativada, se ela não for mais necessária, exclua-a.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Escolha o nome do usuário para ir para a página de detalhes do usuário.
4. Na guia Credenciais de segurança, na seção Chaves de acesso, escolha o menu suspenso Ações para a chave de acesso inativa e, em seguida, Excluir.
5. Na caixa de diálogo Excluir, confirme que você deseja excluir a chave de acesso inserindo o ID da chave de acesso no campo de entrada de texto e, em seguida, selecionando Excluir.

Depois que uma chave de acesso é excluída, ela não pode ser recuperada.

AWS CLI

Execute este comando:

- `aws iam delete-access-key`

API

Chame a seguinte operação:

- `DeleteAccessKey`

Para listar as chaves de acesso de um usuário do IAM

Você pode visualizar uma lista dos IDs de chave de acesso associados a um usuário do IAM.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Escolha o nome do usuário para ir para a página de detalhes do usuário.
4. Na guia Credenciais de segurança, a seção Chaves de acesso lista os IDs de chave de acesso do usuário, incluindo o status de cada chave exibida.

   > **Note**
   >
   > Somente o ID de chave de acesso do usuário é visível. A chave de acesso secreta só pode ser recuperada quando a chave é criada.

   Cada usuário do IAM pode ter duas chaves de acesso.

AWS CLI

Execute este comando:

- `aws iam list-access-keys`

API

Chame a seguinte operação:

- `ListAccessKeys`

Para exibir todos os IDs de chave de acesso dos usuários na conta

É possível visualizar uma lista dos IDs de chave de acesso dos usuários na Conta da AWS.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Escolha o nome do usuário para ir para a página de detalhes do usuário.
4. Se necessário, adicione a coluna Access key ID à tabela de usuários concluindo as etapas a seguir:

   a. Acima da tabela, na extrema direita, selecione o ícone de Preferências ( ).

   b. Na caixa de diálogo Preferências, em Selecionar colunas visíveis, ative o ID da chave de acesso.

   c. Escolha Confirmar para retornar à lista de usuários. A lista é atualizada para incluir o ID da chave de acesso.

5. A coluna ID da chave de acesso mostra o estado de cada chave de acesso, seguido de seu ID; por exemplo, **`Active - AKIAIOSFODNN7EXAMPLE`** ou **`Inactive - AKIAI44QH8DHBEXAMPLE`**.

   Você pode usar essas informações para visualizar e copiar os IDs das chaves de acesso para usuários com uma ou duas chaves de acesso. A coluna exibe **`-`** para usuários sem chave de acesso.

   > **Note**
   >
   > A chave de acesso secreta só pode ser recuperada quando a chave é criada.

   Cada usuário do IAM pode ter duas chaves de acesso.

Para usar um ID de chave de acesso para localizar um usuário

Você pode usar um ID de chave de acesso para localizar um usuário na Conta da AWS.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, na caixa de pesquisa, insira o ID da chave de acesso. Por exemplo, AKIAI44QH8DHBEXAMPLE.
3. O usuário do IAM ao qual o ID da chave de acesso está associado aparece no painel de navegação. Escolha o nome do usuário para ir para a página de detalhes do usuário.

Para localizar o uso mais recente de um ID de chave de acesso

O uso mais recente de uma chave de acesso é exibido na lista do usuário na página de usuários do IAM, na página de detalhes do usuário, e faz parte do relatório de credenciais.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Na lista de usuários, consulte a coluna Chave de acesso usada pela última vez.

   Se a coluna não for exibida, clique no ícone Preferences (Preferências) ( ) e, em Select visible columns (Selecionar colunas visíveis), ative a opção Access key last used (Última chave de acesso utilizada) para exibir a coluna.
3. (opcional) No painel de navegação, em Relatórios de acesso, selecione Relatório de credenciais para fazer download de um relatório que inclua as informações da última chave de acesso usada para todos os usuários do IAM na conta.
4. (opcional) Selecione o usuário do IAM para visualizar os detalhes de usuário. A seção Resumo inclui os IDs das chaves de acesso, o status e quando foram usadas pela última vez.

AWS CLI

Execute este comando:

- `aws iam get-access-key-last-used`

API

Chame a seguinte operação:

- `GetAccessKeyLastUsed`

Atualizar chaves de acesso

Como melhor prática de segurança, sugerimos atualizar as chaves de acesso do usuário do IAM quando necessário, por exemplo, quando um funcionário sair da sua empresa. Os usuários do IAM poderão atualizar suas próprias chaves de acesso se tiverem recebido as permissões necessárias.

Para obter detalhes sobre a concessão de permissões aos usuários do IAM para atualizar suas próprias chaves de acesso, consulte AWS: permite que os usuários do IAM gerenciem suas próprias senhas, chaves de acesso e chaves públicas SSH na página Credenciais de segurança. Você também pode aplicar uma política de senha à sua conta para exigir que todos os seus usuários do IAM atualizem suas senhas periodicamente e definir a frequência com a qual eles devem fazer isso. Para obter mais informações, consulte Definir uma política de senhas de contas para usuários do IAM.

> **Note**
>
> Se tiver perdido a chave de acesso secreta, você deverá excluir a chave de acesso e criar uma nova. A chave de acesso secreta pode ser recuperada somente no momento em que você a cria. Use esse procedimento para desativar e substituir chaves de acesso perdidas por novas credenciais.

Tópicos

- Atualização das chaves de acesso do usuário do IAM (console)
- Atualização das chaves de acesso (AWS CLI)
- Atualização de chaves de acesso (API da AWS)

Atualização das chaves de acesso do usuário do IAM (console)

É possível atualizar as chaves de acesso via Console de gerenciamento da AWS.

Para atualizar as chaves de acesso de um usuário do IAM sem interromper suas aplicações (console)

1. Embora a primeira chave de acesso ainda esteja ativa, crie uma segunda chave de acesso.

   a. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.

   b. No painel de navegação, escolha Usuários.

   c. Selecione o nome do usuário desejado e, em seguida, selecione a guia Credenciais de segurança.

   d. Na seção Chaves de acesso, escolha Criar chave de acesso. Na página Access key best practices & alternatives (Práticas recomendadas e alternativas da chave de acesso), escolha Other (Outro) e escolha Next (Avançar).

   e. (Opcional) Defina um valor de etiqueta de descrição para a chave de acesso para adicionar um par chave-valor de etiqueta a esse usuário do IAM. Isso pode ajudar a identificar e a atualizar as chaves de acesso posteriormente. A chave da etiqueta é definida como o ID da chave de acesso. O valor da etiqueta é definido com a descrição da chave de acesso que você especifica. Quando terminar, escolha Create access key (Criar chave de acesso).

   f. Na página Retrieve access keys (Recuperar chaves de acesso), escolha Show (Exibir) para revelar o valor da chave de acesso secreta do usuário ou Download .csv file (Baixar o arquivo.csv). Essa é a única oportunidade de salvar a chave de acesso secreta. Depois de salvar a chave de acesso secreta em um local seguro, escolha Done (Concluído).

      Quando você cria uma chave de acesso para seu usuário, o par de chaves é ativo por padrão, e você pode usar o par imediatamente. Neste momento, o usuário terá duas chaves de acesso ativas.

2. Atualize todos os aplicativos e ferramentas para usarem a nova chave de acesso.
3. Determine se a primeira chave de acesso ainda está em uso, analisando a informação Last used (Usado pela última vez) da chave de acesso mais antiga. Uma maneira é esperar vários dias e depois verificar se a chave de acesso antiga foi usada antes de prosseguir.
4. Mesmo se a informação Last used (Usada pela última vez) indicar que a chave antiga nunca foi usada, recomendamos que você não exclua imediatamente a primeira chave de acesso. Em vez disso, escolha Actions (Ações) e escolha Deactivate (Desativar) para desativar a primeira chave de acesso.
5. Use somente a nova chave de acesso para confirmar que seus aplicativos estão funcionando. Todos os aplicativos e ferramentas que ainda usarem a chave de acesso original deixarão de

funcionar nesse momento, pois eles não terão mais acesso aos recursos da AWS. Se encontrar uma aplicação ou ferramenta nessa situação, você poderá reativar a primeira chave de acesso. Em seguida, retorne a Step 3 e atualize esse aplicativo para usar a nova chave.

6. Depois de aguardar um período para certificar-se de que todos os aplicativos e as ferramentas sejam atualizados, você poderá excluir a primeira chave de acesso:

   a. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.

   b. No painel de navegação, escolha Usuários.

   c. Selecione o nome do usuário desejado e, em seguida, selecione a guia Credenciais de segurança.

   d. Na seção Access keys (Chaves de acesso) da chave de acesso que deseja excluir, escolha Actions (Ações) e escolha Delete (Excluir). Siga as instruções na caixa de diálogo para primeiro Deactivate (Desativar) e depois confirme a exclusão.

Para determinar quais chaves de acesso precisam ser atualizadas ou excluídas (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Se necessário, adicione a coluna Idade da chave de acesso à tabela de usuários concluindo as etapas a seguir:

   a. Acima da tabela, no canto direito, selecione o ícone de configurações ( ).

   b. Em Gerenciar colunas, selecione Idade da chave de acesso.

   c. Escolha Fechar para retornar à lista de usuários.
4. A coluna Idade da chave de acesso mostra o número de dias desde que a chave de acesso ativa mais antiga foi criada. É possível usar essas informações para encontrar usuários com chaves de acesso que precisem ser atualizadas ou excluídas. A coluna exibe Nenhum para usuários sem chaves de acesso.

Atualização das chaves de acesso (AWS CLI)

É possível atualizar as chaves de acesso via AWS Command Line Interface.

Para atualizar chaves de acesso sem interromper suas aplicações (AWS CLI)

1. Embora a primeira chave de acesso ainda esteja ativa, crie uma segunda chave de acesso, que é ativada por padrão. Execute o seguinte comando:

   - `aws iam create-access-key`

   Neste momento, o usuário terá duas chaves de acesso ativas.
2. Atualize todos os aplicativos e ferramentas para usarem a nova chave de acesso.
3. Determinar se a primeira chave de acesso ainda está em uso por meio deste comando:

   - `aws iam get-access-key-last-used`

   Uma maneira é esperar vários dias e depois verificar se a chave de acesso antiga foi usada antes de prosseguir.
4. Mesmo se etapa Step 3 indicar que não há uso da chave antiga, recomendamos que você não exclua imediatamente a primeira chave de acesso. Em vez disso, altere o estado da primeira chave de acesso para `Inactive` usando este comando:

   - `aws iam update-access-key`
5. Use somente a nova chave de acesso para confirmar que seus aplicativos estão funcionando. Todos os aplicativos e ferramentas que ainda usarem a chave de acesso original deixarão de funcionar nesse momento, pois eles não terão mais acesso aos recursos da AWS. Se encontrar uma aplicação ou ferramenta nessa situação, alterne o estado para `Active` para reativar a primeira chave de acesso. Em seguida, retorne à etapa Step 2 e atualize esse aplicativo para usar a nova chave.
6. Depois de aguardar um período para certificar-se de que todos os aplicativos e as ferramentas sejam atualizados, você poderá excluir a primeira chave de acesso com este comando:

   - `aws iam delete-access-key`

Atualização de chaves de acesso (API da AWS)

É possível atualizar chaves de acesso usando a API da AWS.

Para atualizar chaves de acesso sem interromper suas aplicações (API da AWS)

1. Embora a primeira chave de acesso ainda esteja ativa, crie uma segunda chave de acesso, que é ativada por padrão. Chame a seguinte operação:

   - `CreateAccessKey`

   Neste momento, o usuário terá duas chaves de acesso ativas.
2. Atualize todos os aplicativos e ferramentas para usarem a nova chave de acesso.
3. Determinar se a primeira chave de acesso ainda está em uso chamando esta operação:

   - `GetAccessKeyLastUsed`

   Uma maneira é esperar vários dias e depois verificar se a chave de acesso antiga foi usada antes de prosseguir.
4. Mesmo se etapa Step 3 indicar que não há uso da chave antiga, recomendamos que você não exclua imediatamente a primeira chave de acesso. Em vez disso, altere o estado da primeira chave de acesso para `Inactive` chamando esta operação:

   - `UpdateAccessKey`
5. Use somente a nova chave de acesso para confirmar que seus aplicativos estão funcionando. Todos os aplicativos e ferramentas que ainda usarem a chave de acesso original deixarão de funcionar nesse momento, pois eles não terão mais acesso aos recursos da AWS. Se encontrar uma aplicação ou ferramenta nessa situação, alterne o estado para `Active` para reativar a primeira chave de acesso. Em seguida, retorne à etapa Step 2 e atualize esse aplicativo para usar a nova chave.
6. Depois de aguardar um período para certificar-se de que todas as aplicações e as ferramentas sejam atualizadas, você poderá excluir a primeira chave de acesso chamando esta operação:

   - `DeleteAccessKey`

Proteger chaves de acesso

Qualquer pessoa que tem suas chaves de acesso possui o mesmo nível de acesso a seus recursos da AWS que você tiver. Consequentemente, a AWS executa vários procedimentos para proteger suas chaves de acesso e, de acordo com o nosso modelo de responsabilidade compartilhada, você deve fazer o mesmo.

Expanda as seções a seguir para obter orientação sobre como proteger suas chaves de acesso.

> **Note**
>
> Sua organização pode ter requisitos e políticas de segurança diferentes dos descritos neste tópico. As sugestões fornecidas aqui devem ser usadas como diretrizes gerais.

Remova (ou não gere) as chaves de acesso de Usuário raiz da conta da AWS

Uma das melhores maneiras de proteger a sua conta é não ter chaves de acesso para seu Usuário raiz da conta da AWS. A não ser que você tenha que ter chaves de acesso do usuário raiz (algo que é raro), é melhor não criá-las. Em vez disso, crie um usuário administrativo Centro de Identidade do AWS IAM para as tarefas administrativas diárias. Para obter informações sobre como criar um usuário administrativo no Centro de Identidade do IAM, consulte Introdução no Guia do usuário do Centro de Identidade do IAM.

Se você já tiver uma chave de acesso para o usuário raiz, recomendamos o seguinte: encontre lugares em suas aplicações onde você usa chaves de acesso (se houver) e substitua a chave de acesso do usuário raiz por chaves de acesso de usuário do IAM. Em seguida, desabilite e remova as chaves de acesso do usuário raiz. Para obter mais informações sobre como atualizar as chaves de acesso, consulte Atualizar chaves de acesso

Usar credenciais de segurança temporárias (perfis do IAM) em vez de chaves de acesso de longo prazo

Em muitos casos, você não precisa de chaves de acesso de longo prazo que nunca expiram (como há no caso de um usuário do IAM). Em vez disso, você pode criar perfis do IAM e gerar credenciais de segurança temporárias. As credenciais de segurança temporárias consistem em um ID da chave de acesso e uma chave de acesso secreta, mas elas também incluem um token de segurança que indica quando as credenciais expiram.

As chaves de acesso de longo prazo, como aquelas associadas a contas de usuários do IAM e ao usuário raiz, permanecem válidas até serem revogadas manualmente. No entanto, as credenciais de segurança temporárias obtidas por meio de perfis do IAM e outros recursos do AWS Security Token Service expiram após um curto período. Use credenciais de segurança temporárias para ajudar a reduzir os riscos em caso de credenciais que sejam expostas acidentalmente.

Use um perfil do IAM e credenciais de segurança temporárias nestes cenários:

- Você tem uma aplicação ou scripts da AWS CLI em execução em uma instância do Amazon EC2. Não use chaves de acesso diretamente em sua aplicação. Não passe uma chave de acesso para a aplicação, não incorpore-a na aplicação nem deixe que a aplicação leia chaves de acesso de nenhuma fonte. Em vez disso, defina um perfil do IAM que tenha as permissões apropriadas para sua aplicação e execute a instância do Amazon Elastic Compute Cloud (Amazon EC2) com perfis para o EC2. Fazer isso associará um perfil do IAM à instância do Amazon EC2. Essa prática também permite que a aplicação obtenha credenciais de segurança temporárias que ela pode, por sua vez, usar para fazer chamadas programáticas para a AWS. Os SDKs da AWS e a AWS Command Line Interface (AWS CLI) podem obter credenciais temporárias do perfil automaticamente.
- Você precisa conceder acesso entre contas. Use um perfil do IAM para estabelecer confiança entre contas e, em seguida, conceda aos usuários em uma conta permissões limitadas para acessar a conta confiável. Para obter mais informações, consulte Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM.
- Você tem um aplicativo móvel. Não integre chaves de acesso à aplicação, mesmo em armazenamento criptografado. Em vez disso, use o Amazon Cognito para gerenciar a identidade do usuário na sua aplicação. Esse serviço permite autenticar os usuários usando login com o Amazon, Facebook, Google ou qualquer provedor de identidade compatível com o OpenID Connect (OIDC). Em seguida, é possível usar o provedor de credenciais do Amazon Cognito para gerenciar credenciais que sua aplicação use para fazer solicitações à AWS.
- Você deseja centralizar-se na AWS e sua organização tem suporte para SAML 2.0. Se você trabalha para uma organização que tenha um provedor de identidade compatível com o SAML 2.0, configure o provedor para usar o SAML. É possível usar o SAML para trocar informações de autenticação com a AWS e obter novamente um conjunto de credenciais de segurança temporárias. Para obter mais informações, consulte Federação SAML 2.0.
- Você deseja se federar na AWS e sua organização tem um armazenamento de identidades on-premises. Se os usuários podem autenticar dentro da sua organização, você poderá escrever um aplicativo que emite a eles credenciais de segurança temporárias para acesso a recursos da AWS. Para obter mais informações, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.
- Use condições nas políticas do IAM para permitir somente o acesso das redes esperadas. Você pode limitar onde e como suas chaves de acesso são usadas ao implementar políticas do IAM com condições que especificam e permitem somente redes esperadas, como seus endereços IP públicos ou nuvens privadas virtuais (VPCs). Dessa forma, você sabe que as chaves de acesso só poderão ser usadas em redes esperadas e aceitáveis.

> **Note**
>
> Você está usando uma instância do Amazon EC2 com uma aplicação que precisa de acesso programático a recursos da AWS? Se sim, use perfis do IAM para EC2.

Gerenciar chaves de acesso de usuário do IAM corretamente

Se for necessário criar chaves de acesso para acesso programático à AWS, crie-as para usuários do IAM, concedendo aos usuários somente as permissões necessárias.

Observe estas precauções para ajudar a proteger as chaves de acesso de usuários do IAM:

- Não incorpore chaves de acesso diretamente no código. Os SDKs da AWS e as Ferramentas da linha de comando da AWS permitem colocar chaves de acesso em pontos conhecidos para que você não precise mantê-las no código.

  Coloque chaves de acesso em um dos seguintes locais:

  - O arquivo de credenciais da AWS. Os SDKs da AWS e a AWS CLI usam automaticamente as credenciais que você armazena no arquivo de credenciais da AWS.

    Para obter informações sobre como usar o arquivo de credenciais da AWS, consulte a documentação do SDK. Os exemplos incluem Configurar credenciais e região da AWS no Guia do desenvolvedor do AWS SDK para Java e Arquivos de configuração e credenciais no Guia do usuário da AWS Command Line Interface.

    Para armazenar credenciais para o AWS SDK para .NET e o AWS Tools for Windows PowerShell, recomendamos que você use a SDK Store. Para obter mais informações, consulte Utilização da SDK Store no Guia do desenvolvedor do AWS SDK para .NET.
  - Variáveis de ambiente. Em um sistema multicliente, escolha as variáveis de ambiente do usuário, não as variáveis de ambiente do sistema.

    Para obter mais informações sobre o uso de variáveis de ambiente para armazenar credenciais, consulte Variáveis de ambiente no Guia do usuário da AWS Command Line Interface.
- Use chaves de acesso diferentes para aplicativos diferentes. Faça isso para que seja possível isolar as permissões e revogar as chaves de acesso para aplicações específicas caso elas sejam expostas. Ter chaves de acesso separadas para diferentes aplicativos também gera entradas distintas em arquivos de log do AWS CloudTrail. Essa configuração facilita a determinação de qual aplicativo executou ações específicas.

- Atualize as chaves de acesso quando necessário. Se houver risco de comprometimento da chave de acesso, atualize a chave de acesso e exclua a chave de acesso anterior. Para obter detalhes, consulte Atualizar chaves de acesso
- Remover chaves de acesso não utilizadas. Se um usuário sair da sua organização, remova o usuário do IAM correspondente para que ele não possa mais acessar seus recursos. Para saber quando uma chave de acesso foi usada pela última vez, use a API `GetAccessKeyLastUsed` (comando da AWS CLI: `aws iam get-access-key-last-used`).
- Use credenciais temporárias e configure a autenticação multifator para suas operações de API mais confidenciais. Com as políticas do IAM, você pode especificar quais operações de API um usuário tem permissão para chamar. Em alguns casos, talvez você deseje obter segurança adicional para exigir que os usuários sejam autenticados com a MFA da AWS antes que eles tenham permissão para executar ações particularmente confidenciais. Por exemplo, você pode ter uma política que permita que um usuário execute as ações `RunInstances`, `DescribeInstances` e `StopInstances` do Amazon EC2. Porém, você pode restringir uma ação destrutiva, tal como `TerminateInstances` e certificar-se de que os usuários possam executar essa ação apenas se eles autenticarem com um dispositivo MFA AWS. Para obter mais informações, consulte Acesso seguro à API com a MFA.

Acessar o aplicativo móvel usando chaves de acesso da AWS

Você pode acessar um conjunto limitado de serviços e recursos da AWS usando o aplicativo móvel AWS. O aplicativo móvel ajuda a oferecer suporte à resposta a incidentes em trânsito. Para obter mais informações e fazer download do aplicativo, consulte Aplicativo móvel do console da AWS.

Você pode fazer login no aplicativo móvel usando sua senha do console ou suas chaves de acesso. Como prática recomendada, não use chaves de acesso de usuário raiz. Em vez disso, recomendamos enfaticamente que, além de utilizar uma senha ou um acesso biométrico no seu dispositivo móvel, criar um usuário do IAM especificamente para gerenciar os recursos da AWS usando a aplicação móvel. Caso você perca dispositivo móvel, poderá remover o acesso do usuário do IAM.

Para fazer login usando chaves de acesso (aplicativo móvel)

1. Abra o aplicativo no dispositivo móvel.
2. Se esta for a primeira vez que você adiciona uma identidade ao dispositivo, escolha Add an identity (Adicionar uma identidade) e selecione Access keys (Chaves de acesso).

Se você já fez login usando outra identidade, escolha o ícone de menu e selecione Switch identity (Alternar identidade). Depois escolha Sign in as a different identity (Fazer login como uma identidade diferente) e Access keys (Chaves de acesso).

3. Na página Access keys (Chaves de acesso), insira suas informações:

   - ID da chave de acesso: insira o ID da sua chave de acesso.
   - Chave de acesso secreta: insira sua chave de acesso secreta.
   - Nome da identidade: insira o nome da identidade que aparecerá na aplicação móvel. Ele não precisa ser igual ao seu nome de usuário do IAM.
   - PIN de identidade: crie um número de identificação pessoal (PIN) que você usará para futuros logins.

   > **Note**
   >
   > Se habilitar a biometria para o aplicativo móvel da AWS, você será solicitado a usar sua impressão digital ou o reconhecimento facial para verificação em vez do PIN. Se a biometria falhar, você poderá ser solicitado a fornecer o PIN.

4. Escolha Verify and add keys (Verificar e adicionar chaves).

   Agora você pode acessar um conjunto selecionado dos seus recursos usando o aplicativo móvel.

Informações relacionadas

Os tópicos a seguir fornecem diretrizes para a configuração dos SDKs da AWS e da AWS CLI para o uso das chaves de acesso.

- Defina as credenciais e a região da AWS no Guia do desenvolvedor do AWS SDK para Java
- Utilização da SDK Store no Guia do desenvolvedor do AWS SDK para .NET
- Fornecimento de credenciais para o SDK no Guia do desenvolvedor do AWS SDK para PHP
- Configuração na documentação do Boto 3 (SDK da AWS para Python)
- Uso de credenciais da AWS no Guia do usuário do AWS Tools for Windows PowerShell
- Arquivos de configuração e credenciais no Guia do usuário da AWS Command Line Interface
- Concessão de acesso usando um perfil do IAM no Guia do desenvolvedor do AWS SDK para .NET

- Configuração de perfis do IAM para o Amazon EC2 no AWS SDK for Java 2.x

Usar credenciais da chave secreta e chave de acesso para acesso ao console

É possível usar credenciais da chave secreta e chave de acesso para acesso direto ao Console de gerenciamento da AWS, não apenas à AWS CLI. Isso pode ser feito usando a chamada de API `GetFederationToken` do AWS STS. Ao criar um URL de console usando as credenciais temporárias e o token fornecidos pelo `GetFederationToken`, as entidades principais do IAM podem acessar o console. Para obter mais informações, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.

Vale esclarecer que, ao fazer login no console diretamente usando credenciais do IAM ou de usuário-raiz com o MFA ativado, o MFA será necessário. No entanto, se o método descrito acima (usar credenciais temporárias com `GetFederationToken`) for usado, o MFA NÃO será necessário.

Fazer auditoria de chaves de acesso

É possível revisar as chaves de acesso da AWS em seu código para determinar se as chaves são de uma conta que você possui. É possível transmitir um ID de chave de acesso usando o comando `aws sts get-access-key-info` da AWS CLI ou a operação da API `GetAccessKeyInfo` da AWS.

As operações de API da AWS e da AWS CLI retornam o ID da Conta da AWS à qual a chave de acesso pertence. Os IDs de chave de acesso que começam com `AKIA` são credenciais de longo prazo para um usuário do IAM ou um Usuário raiz da conta da AWS. Os IDs de chave de acesso que começam com `ASIA` são credenciais temporárias que são criadas usando operações do AWS STS. Se a conta na resposta pertencer a você, você poderá fazer login como usuário raiz e revisar suas chaves de acesso de usuário raiz. Em seguida, você pode obter um relatório de credenciais para saber qual usuário do IAM é o proprietário das chaves. Para saber quem solicitou as credenciais temporárias para uma chave de acesso `ASIA`, visualize os eventos do AWS STS nos logs do CloudTrail.

Por motivos de segurança, você pode revisar logs do AWS CloudTrail para saber quem executou uma ação na AWS. Você pode usar a chave de condição `sts:SourceIdentity` na política de confiança da função para exigir que os usuários especifiquem uma identidade quando assumirem uma função. Por exemplo, você pode exigir que os usuários do IAM especifiquem seu próprio nome de usuário como a identidade-fonte. Isso pode ajudar você a determinar qual usuário executou uma ação específica na AWS. Para obter mais informações, consulte `sts:SourceIdentity`.

Essa operação não indica o estado da chave de acesso. A chave pode estar ativa, inativa ou excluída. As chaves ativas podem não ter permissões para executar uma operação. Fornecer uma chave de acesso excluída pode retornar um erro informando que a chave não existe.

AWSCódigo da autenticação multifator no IAM da

Para obter mais segurança, recomendamos que você configure a autenticação multifator (MFA) para ajudar a proteger seus recursos da AWS. Você pode habilitar a MFA para os Usuário raiz da conta da AWS de todas as Contas da AWS, incluindo contas autônomas, contas de gerenciamento e contas de membros, bem como para os usuários do IAM. Sempre que possível, recomendamos usar MFA resistente a phishing, como chaves de acesso e chaves de segurança. Esses autenticadores baseados em FIDO usam criptografia de chave pública e são resistentes a ataques de phishing, man-in-the-middle e replay, fornecendo um nível de segurança mais forte do que as opções baseadas em TOTP.

A MFA é aplicada a todos os tipos de conta do usuário-raiz. Para obter mais informações, consulte Proteja as credenciais de usuário-raiz da sua conta do AWS Organizations.

Quando você habilitar a MFA para o usuário raiz, ela afetará somente as credenciais do usuário raiz. Os usuários do IAM na conta são identidades distintas com suas próprias credenciais, e cada identidade tem sua própria configuração de MFA. Para obter mais informações sobre como usar a MFA para proteger o usuário-raiz, consulte Autenticação multifator para Usuário raiz da conta da AWS.

Seu Usuário raiz da conta da AWS e usuários do IAM podem registrar até oito dispositivos MFA de qualquer tipo. O registro de vários dispositivos de MFA pode oferecer flexibilidade e ajudar a reduzir o risco de interrupção do acesso se um dispositivo for perdido ou quebrado. Basta um dispositivo MFA para acessar o Console de gerenciamento da AWS ou criar uma sessão pela AWS CLI.

> **Note**
>
> Recomendamos exigir que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Você já pensou em usar o Centro de Identidade do AWS IAM? O IAM Identity Center pode ser usado para gerenciar centralmente o acesso a várias Contas da AWS e fornecer aos usuários acesso de logon único protegido por MFA a todas as contas atribuídas em um só lugar. Com o IAM Identity Center, é possível criar e gerenciar identidades de usuários no IAM Identity Center ou conectar facilmente ao provedor de

identidades compatível com SAML 2.0 existente. Para obter mais informações, consulte O que é o IAM Identity Center? no Guia do usuário do Centro de Identidade do AWS IAM.

A MFA adiciona segurança extra que exige que os usuários forneçam autenticação exclusiva de um mecanismo de MFA compatível com a AWS, além de credenciais de login, quando acessam sites ou serviços da AWS.

Tipos de MFA

AWSA é compatível com os seguintes tipos de MFA:

Sumário

- Chaves de acesso e chaves de segurança
- Aplicações de autenticador virtual
- Tokens físicos de TOTP

Chaves de acesso e chaves de segurança

AWS Identity and Access ManagementA é compatível com chaves de acesso e chaves de segurança para MFA. Com base nos padrões FIDO, chaves de acesso usam criptografia de chave pública para proporcionar uma autenticação forte e resistente a phishing que é mais segura do que as senhas. A AWS é compatível com dois tipos de chaves de acesso: chaves de acesso vinculadas a dispositivo (chaves de segurança) e chaves de acesso sincronizadas.

- Chaves de segurança: dispositivos físicos, como a YubiKey, usados como segundo fator de autenticação. Uma única chave de segurança é compatível com várias contas de usuário-raiz e usuários do IAM.
- Chaves de acesso sincronizadas: usam gerenciadores de credenciais de provedores como Google, Apple, contas da Microsoft e serviços de terceiros, como 1Password, Dashlane e Bitwarden, como segundo fator.

É possível usar autenticadores biométricos integrados, como Touch ID em Apple MacBooks, para desbloquear seu gerenciador de credenciais e fazer login na AWS. As chaves de acesso são criadas com o provedor escolhido usando sua impressão digital, rosto ou PIN do dispositivo. É possível usar uma chave de acesso de autenticação entre dispositivos (CDA) de um dispositivo, como um

dispositivo móvel ou uma chave de segurança de hardware, para entrar em outro dispositivo, como um laptop. Para obter mais informações, consulte autenticação entre dispositivos (CDA).

É possível sincronizar chaves de acesso em seus dispositivos para facilitar o login na AWS e aprimorar a usabilidade e a capacidade de recuperação. Para obter mais informações sobre como habilitar chaves de acesso e chaves de segurança, consulte Habilitar uma chave de acesso ou uma chave de segurança para o usuário-raiz (console).

A FIDO Alliance conta com uma lista de todos os produtos certificados pela FIDO compatíveis com as especificações da FIDO.

Aplicações de autenticador virtual

Uma aplicação de autenticador virtual funciona em um telefone ou outro dispositivo e emula um dispositivo físico. As aplicações de autenticação virtual implementam o algoritmo de senha de uso único com marcação temporal (TOTP) e oferecem suporte a vários tokens em um único dispositivo. O usuário deve digitar um código válido no dispositivo quando solicitado durante o processo de acesso. Cada token atribuído a um usuário deve ser exclusivo. O usuário não pode digitar um código do token de outro usuário para se autenticar.

Recomendamos usar MFA resistente a phishing, como chaves de acesso ou chaves de segurança, para obter a proteção mais forte. Se ainda não for possível usar chaves de acesso ou chaves de segurança, recomendamos usar um dispositivo MFA virtual como medida provisória enquanto aguarda a aprovação da compra do hardware ou aguarda a chegada do hardware. Para obter uma lista de algumas aplicações compatíveis que podem ser usadas como dispositivos de MFA virtuais, consulte Autenticação multifator (MFA).

Para obter instruções sobre como configurar um dispositivo virtual de MFA para um usuário do IAM, consulte Atribuir um dispositivo MFA virtual ao Console de gerenciamento da AWS.

> **Note**
>
> Os dispositivos virtuais de MFA não atribuídos na Conta da AWS são excluídos quando você adiciona novos dispositivos virtuais de MFA por meio do Console de gerenciamento da AWS ou durante o processo de login. Dispositivos virtuais de MFA não atribuídos são dispositivos na conta, mas que não são usados pelo usuário-raiz da conta ou pelos usuários do IAM no processo de login. Eles são excluídos para que novos dispositivos virtuais de MFA possam ser adicionados à conta. Também é possível reutilizar nomes de dispositivos.

- Para visualizar dispositivos virtuais de MFA não atribuídos na conta, você pode usar o comando list-virtual-mfa-devices da AWS CLI ou a chamada de API.
- Para desativar um dispositivo virtual de MFA, você pode usar o comando deactivate-mfa-device da AWS CLI ou a chamada de API. O dispositivo ficará sem atribuição.
- Para anexar um dispositivo virtual de MFA não atribuído ao usuário-raiz da Conta da AWS ou aos usuários do IAM, você precisará do código de autenticação gerado pelo dispositivo junto com o comando enable-mfa-device da AWS CLI ou a chamada de API.

Tokens físicos de TOTP

Um dispositivo físico gera um código numérico de seis dígitos baseado no algoritmo de senha de uso único com marcação temporal (TOTP). O usuário deve digitar um código válido no dispositivo em uma segunda página da web durante o login.

Esses tokens são usados exclusivamente com Contas da AWS. Você só pode usar tokens que compartilhem as sementes de token com a de modo seguro AWS. Sementes de token são chaves secretas geradas no momento da produção dos tokens. Tokens comprados de outras fontes não funcionam com o IAM. Para garantir compatibilidade, você deve comprar seu dispositivo físico de MFA em um dos seguintes links: token de OTP ou cartão com visor de OTP.

- Cada dispositivo MFA atribuído a um usuário deve ser exclusivo. Um usuário não pode digitar um código do dispositivo de outro usuário para ser autenticado. Para obter informações sobre os dispositivos físicos de MFA compatíveis, consulte autenticação multifator (MFA).
- Se você quiser usar um dispositivo físico de MFA, recomendamos que use chaves de segurança como alternativa aos dispositivos físicos de TOTP. As chaves de segurança não precisam de bateria, são resistentes a phishing e oferecem suporte a vários usuários em um único dispositivo.

Você só pode habilitar uma chave de acesso ou uma chave de segurança no Console de gerenciamento da AWS, não na AWS CLI ou na API da AWS. Antes que possa habilitar uma chave de segurança, você deve ter acesso físico ao dispositivo.

Para obter instruções sobre como configurar um token físico de TOTP para um usuário do IAM, consulte Atribuir um token de hardware TOTP ao Console de gerenciamento da AWS.

> **Note**
>
> MFA baseada em mensagem de texto de SMS: a AWS não é mais compatível com a habilitação de autenticação multifator (MFA) por SMS. Recomendamos que os clientes com usuários do IAM que usem MFA baseada em mensagem de texto por SMS mudem para um dos seguintes métodos alternativos: chave de acesso ou chave de segurança, dispositivo virtual de MFA (baseado em software) ou dispositivo físico de MFA. Você pode identificar os usuários da sua conta com um dispositivo de MFA por SMS atribuído. No console do IAM, selecione Users (Usuários) no painel de navegação e procure os usuários com SMS na coluna MFA da tabela.

Recomendações para MFA

Para ajudar a proteger suas identidades da AWS, siga as recomendações a seguir para autenticação MFA.

- Recomendamos usar MFA resistente a phishing, como chaves de acesso e chaves de segurança, como dispositivo MFA. Esses autenticadores baseados em FIDO oferecem a proteção mais forte contra ataques como phishing.
- Recomendamos habilitar vários dispositivos com MFA para o Usuário raiz da conta da AWS e usuários do IAM em suas Contas da AWS. Isso permite aumentar o nível de segurança das Contas da AWS e simplificar o gerenciamento do acesso a usuários altamente privilegiados, como o Usuário raiz da conta da AWS.
- Você pode registrar até oito dispositivos com MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu Usuário raiz da conta da AWS e usuários do IAM. Com vários dispositivos de MFA, basta um dispositivo de MFA para acessar o Console de gerenciamento da AWS ou criar uma sessão pela AWS CLI como esse usuário. Um usuário do IAM deve se autenticar com um dispositivo de MFA existente para habilitar ou desabilitar um dispositivo de MFA adicional.
- Caso o dispositivo de MFA seja perdido, roubado ou esteja inacessível, você pode usar um dos dispositivos de MFA restantes para acessar a Conta da AWS sem realizar o procedimento de recuperação de Conta da AWS. Se um dispositivo de MFA for perdido ou roubado, ele deverá ser desassociado da entidade principal do IAM ao qual está associado.

- O uso de vários MFA permite que seus funcionários em locais geograficamente distribuídos ou trabalhando remotamente usem MFA baseada em hardware para acessar a AWS sem precisar coordenar a troca física de um único dispositivo de hardware entre funcionários.
- O uso de dispositivos MFA adicionais para entidades principais do IAM permite que você use um ou mais MFAs para uso diário, além de manter os dispositivos físicos de MFA em um local físico seguro, como um cofre, para fins de backup e redundância.

> Observações
>
> - Você não pode transmitir as informações de MFA de uma chave de segurança ou chave de acesso para operações de API do AWS STS para solicitar credenciais temporárias. É possível obter credenciais para uso com a AWS CLI e os AWS SDKs ao usar uma chave de segurança ou chave de acesso executando o comando `aws login`.
> - Você não pode usar comandos da AWS CLI ou operações de API da AWS para habilitar chaves de segurança FIDO.
> - Não é possível usar o mesmo nome para mais de um usuário-raiz ou MFA do IAM.

Recursos adicionais

Os recursos a seguir podem ajudar você a saber mais sobre a MFA.

- Para obter mais informações sobre como usar a MFA para acessar a AWS, consulte Login habilitado para MFA.
- Você pode usar o IAM Identity Center para habilitar o acesso seguro com MFA ao portal de acesso da AWS, aos aplicativos integrados do IAM Identity Center e à AWS CLI. Para obter mais informações, consulte Enable MFA in IAM Identity Center.
- Para verificar se o MFA já está habilitado para os usuários, consulte Verificar status da MFA.

Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS

As chaves de acesso são um tipo de dispositivo de autenticação multifator (MFA) que você pode usar para proteger seus recursos da AWS. A AWS é compatível com chaves de acesso

sincronizadas e chaves de acesso vinculadas a dispositivo, também conhecidas como chaves de segurança.

As chaves de acesso sincronizadas permitem que os usuários do IAM acessem suas credenciais de login FIDO em muitos de seus dispositivos, até mesmo novos, sem precisar refazer o registro de todos os dispositivos em todas as contas. As chaves de acesso sincronizadas incluem gerenciadores de credenciais primários, como Google, Apple e Microsoft, e gerenciadores de credenciais de terceiros, como 1Password, Dashlane e Bitwarden, como um segundo fator. Você também pode usar a biometria no dispositivo (p. ex., TouchID ou FaceID) para desbloquear o gerenciador de credenciais escolhido para usar chaves de acesso.

Como alternativa, as chaves de acesso vinculadas a dispositivo são vinculadas a uma chave de segurança FIDO que você conecta a uma porta USB do seu computador e, quando solicitado, toca para concluir com segurança o processo de login. Se você já usar uma chave de segurança FIDO com outros serviços e se ela tiver uma configuração compatível com a AWS (por exemplo, a YubiKey 5 Series da Yubico), também poderá usá-la com a AWS. Caso contrário, será necessário comprar uma chave de segurança FIDO se você quiser usar o WebAuthn for MFA na AWS. Além disso, as chaves de segurança FIDO podem comportar vários usuários do IAM ou raízes no mesmo dispositivo, o que aumenta sua utilidade para proteção da conta. Para especificações e informações sobre aquisição para ambos os tipos de dispositivo, consulte Autenticação multifator.

Você pode registrar até oito dispositivos com MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu Usuário raiz da conta da AWS e usuários do IAM. Com vários dispositivos de MFA, basta um dispositivo de MFA para acessar o Console de gerenciamento da AWS ou criar uma sessão pela AWS CLI como esse usuário. Recomendamos que você registre vários dispositivos de MFA. Por exemplo, você pode registrar um autenticador incorporado e também uma chave de segurança que você mantém em um local fisicamente seguro. Se você não conseguir usar o autenticador integrado, poderá usar sua chave de segurança registrada. Para aplicações autenticadoras, também recomendamos habilitar o atributo de backup ou de sincronização na nuvem para ajudar a evitar a perda de acesso à sua conta caso você perca ou quebre o dispositivo que contém as aplicações autenticadoras.

> **Note**
>
> Recomendamos exigir que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Seus usuários podem federar-se à AWS com um provedor de identidade, onde se autenticam usando suas credenciais corporativas e configurações de MFA. Para gerenciar o acesso à AWS e a aplicações empresariais, recomendamos usar o Centro de

Identidade do IAM. Para obter mais informações, consulte o Guia do usuário do Centro de Identidade do IAM.

Tópicos

- Permissões obrigatórias
- Habilitar uma chave de acesso ou uma chave de segurança para seu próprio usuário do IAM (console)
- Habilitar uma chave de acesso ou uma chave de segurança para outro usuário do IAM (console)
- Substituir uma chave de acesso ou chave de segurança
- Configurações compatíveis com o uso de chaves de acesso e chaves de segurança

Permissões obrigatórias

Para gerenciar uma chave de acesso FIDO para seu próprio usuário do IAM enquanto protege as ações confidenciais relacionadas à MFA, você deve ter as permissões da seguinte política:

Note

Os valores de ARN são valores estáticos e não são um indicador do protocolo usado para registrar o autenticador. Descontinuamos o U2F, então todas as novas implementações usam o WebAuthn.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
```

```
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Habilitar uma chave de acesso ou uma chave de segurança para seu próprio usuário do IAM (console)

Você só pode habilitar uma chave de acesso ou uma chave de segurança para seu próprio usuário do IAM no Console de gerenciamento da AWS, não na AWS CLI ou na API da AWS. Antes que possa habilitar uma chave de segurança, você deve ter acesso físico ao dispositivo.

Para habilitar uma chave de acesso ou uma chave de segurança para seu próprio usuário do IAM (console)

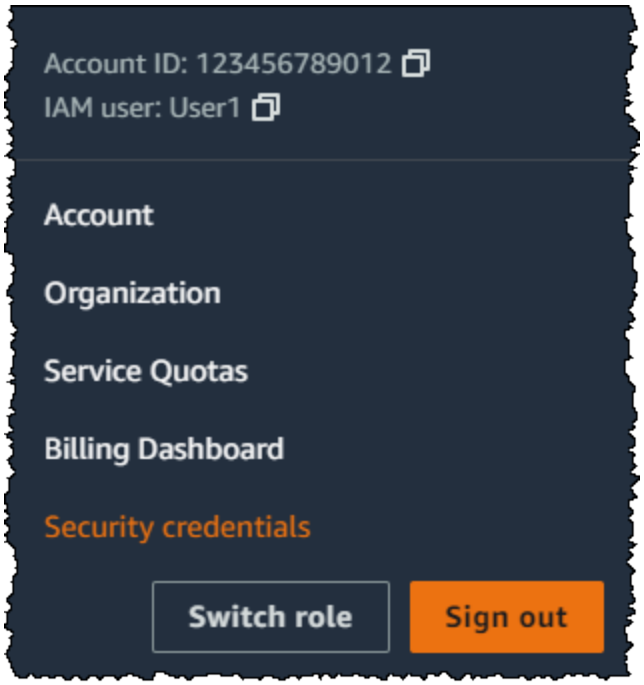
1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

   > **Note**
   >
   > Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login

principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

Para obter o ID da Conta da AWS, fale com o administrador.

2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na página do usuário do IAM selecionado, escolha a guia Credenciais de segurança.
4. Em Multi-Factor Authentication (MFA) (autenticação multifator [MFA]), escolha Assign MFA device (Atribuir dispositivo de MFA).
5. Na página de nome do dispositivo de MFA, insira o nome do dispositivo, escolha Chave de acesso ou Chave de segurança e, em seguida, escolha Avançar.
6. Em Configurar dispositivo, configure sua chave de acesso. Crie uma chave de acesso com dados biométricos, como seu rosto ou impressão digital, com um pin do dispositivo ou inserindo a chave de segurança FIDO na porta USB do seu computador e tocando nela.
7. Siga as instruções no seu navegador e escolha Continuar.

Agora, você registrou sua chave de acesso ou chave de segurança para uso com a AWS. Para obter informações sobre como usar a MFA com o Console de gerenciamento da AWS, consulte Login habilitado para MFA.

Habilitar uma chave de acesso ou uma chave de segurança para outro usuário do IAM (console)

Você só pode habilitar uma chave de acesso ou uma chave de segurança para outro usuário do IAM apenas no Console de gerenciamento da AWS, não na API do AWS CLI ou da AWS.

Para habilitar uma chave de acesso ou de segurança para outro usuário do IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Em Usuários, escolha o nome do usuário para o qual deseja habilitar a MFA.
4. Na página do usuário do IAM selecionado, escolha a guia Credenciais de segurança.
5. Em Multi-Factor Authentication (MFA) (autenticação multifator [MFA]), escolha Assign MFA device (Atribuir dispositivo de MFA).
6. Na página de nome do dispositivo de MFA, insira o nome do dispositivo, escolha Chave de acesso ou Chave de segurança e, em seguida, escolha Avançar.
7. Em Configurar dispositivo, configure sua chave de acesso. Crie uma chave de acesso com dados biométricos, como seu rosto ou impressão digital, com um pin do dispositivo ou inserindo a chave de segurança FIDO na porta USB do seu computador e tocando nela.
8. Siga as instruções no seu navegador e escolha Continuar.

Agora, você registrou uma chave de acesso ou de segurança para outro usuário do IAM usar na AWS. Para obter informações sobre como usar a MFA com o Console de gerenciamento da AWS, consulte Login habilitado para MFA.

Substituir uma chave de acesso ou chave de segurança

Você pode ter até oito dispositivos de MFA com qualquer combinação dos tipos de MFA atualmente compatíveis atribuídos a um usuário ao mesmo tempo com seu Usuário raiz da conta da AWS e usuários do IAM. Se o usuário perder um autenticador compatível com FIDO ou precisar substituí-lo por algum motivo, primeiro você deverá desativar o autenticador FIDO antigo. Em seguida, você poderá adicionar um novo dispositivo MFA para o usuário.

- Para desativar o dispositivo associado no momento a um usuário do IAM, consulte Desativar um dispositivo com MFA.
- Para adicionar uma nova chave de segurança FIDO para um usuário do IAM, consulte Habilitar uma chave de acesso ou uma chave de segurança para seu próprio usuário do IAM (console).

Se você não tiver acesso a uma nova chave de acesso ou uma chave de segurança, poderá habilitar um novo dispositivo virtual de MFA ou um token físico de TOTP. Consulte um dos tópicos a seguir para obter instruções:

- Atribuir um dispositivo MFA virtual ao Console de gerenciamento da AWS
- Atribuir um token de hardware TOTP ao Console de gerenciamento da AWS

Configurações compatíveis com o uso de chaves de acesso e chaves de segurança

Você pode usar as chaves de acesso FIDO2 vinculadas a dispositivo, também conhecidas como chaves de autenticação multifator (MFA), no IAM usando as configurações compatíveis atuais. Dispositivos FIDO2 compatíveis com o IAM e navegadores compatíveis com FIDO2 estão incluídos.

Antes de registrar o dispositivo FIDO2, verifique se você está usando a versão mais recente do navegador e do sistema operacional. O comportamento dos atributos pode ser diferente em diferentes navegadores, autenticadores e clientes de sistema operacional. Se o registro do dispositivo falhar em um navegador, você poderá tentar registrá-lo em outro navegador.

FIDO2 é um padrão aberto de autenticação e uma extensão do FIDO U2F, oferecendo o mesmo alto nível de segurança com base em criptografia de chave pública. FIDO2 consiste na especificação W3C Web Authentication (API do WebAuthn) e no Client-to-Authentication Protocol (CTAP), um protocolo da camada de aplicação. O CTAP permite a comunicação entre cliente ou plataforma, como um navegador ou sistema operacional, e um autenticador externo.

Quando você habilita um autenticador certificado por FIDO na AWS, a chave de segurança FIDO cria um novo par de chaves para uso somente na AWS. Primeiro, você insere suas credenciais. Quando solicitado, você tocará na chave de segurança, que responderá ao desafio de autenticação emitido pela AWS. Para saber mais sobre o padrão FIDO2, consulte Projeto FIDO2.

Dispositivos FIDO2 compatíveis com a AWS

O IAM é compatível com dispositivos de segurança FIDO2 que se conectam aos dispositivos por USB, Bluetooth ou NFC. O IAM também é compatível com os autenticadores de plataforma, como TouchID ou FaceID. O IAM não oferece suporte ao registro de chave de acesso local para o Windows Hello. Para criar e usar chaves de acesso, os usuários do Windows devem usar a autenticação entre dispositivos, na qual você usa uma chave de acesso de um dispositivo, como um dispositivo móvel, ou uma chave de segurança de hardware, para entrar em outro dispositivo, como um laptop.

> **Note**
>
> AWSA precisa ter acesso à porta USB física no computador para verificar o dispositivo U2F. As chaves de segurança não funcionarão com uma máquina virtual, uma conexão remota ou o modo anônimo de um navegador.

A FIDO Alliance mantém uma lista de todos os produtos FIDO2 que são compatíveis com as especificações da FIDO.

Navegadores compatíveis com FIDO2

A disponibilidade dos dispositivos de segurança FIDO2 que são executados em um navegador da Web depende da combinação de navegador e sistema operacional. Os seguintes navegadores são compatíveis com o uso de chaves de segurança:

Navegador da web	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Sim	Sim	Sim	Sim	Não
Safari	Sim	Não	Não	Sim	Não
Edge	Sim	Sim	Não	Sim	Não
Firefox	Sim	Sim	Não	Sim	Não

> **Note**
>
> A maioria das versões do Firefox que são compatíveis com FIDO2 atualmente não habilita a compatibilidade por padrão. Para obter instruções sobre a habilitação da compatibilidade com FIDO2 no Firefox, consulte Solucionar problemas de chaves de acesso e chaves de segurança FIDO.
>
> O Firefox no macOS pode não ser totalmente compatível com fluxos de trabalho de autenticação entre dispositivos para chaves de acesso. Você pode receber um prompt para tocar em uma chave de segurança em vez de prosseguir com a autenticação entre

dispositivos. Recomendamos usar outro navegador, como o Chrome ou o Safari, para fazer login com chaves de acesso no macOS.

Para obter mais informações sobre a compatibilidade do navegador com um dispositivo com certificação FIDO2, como o YubiKey, consulte Operating system and web browser support for FIDO2 and U2F.

Plug-ins de navegador

A AWS só é compatível com navegadores que têm compatibilidade nativa com o padrão FIDO2. A AWS não é compatível com o uso de plug-ins para adicionar compatibilidade com o navegador FIDO2. Alguns plug-ins de navegador são incompatíveis com o padrão FIDO2 e podem causar resultados inesperados com chaves de segurança FIDO2.

Para obter informações sobre como desabilitar plugins do navegador e outras dicas de solução de problemas, consulte Não consigo habilitar minha chave de segurança FIDO.

Certificações de dispositivos

Capturamos e atribuímos certificações relacionadas ao dispositivo, como validação FIPS e nível de certificação FIDO, somente durante o registro de uma chave de segurança. A certificação do seu dispositivo é obtida do Serviço de metadados (MDS) da FIDO Alliance.

Se o status ou o nível de certificação de sua chave de segurança mudar, isso não será refletido automaticamente nas tags do dispositivo. Para atualizar as informações de certificação de um dispositivo, registre-o novamente para buscar as informações de certificação atualizadas.

AWSA fornece os seguintes tipos de certificação como chaves de condição durante o registro do dispositivo, obtidos no FIDO MDS: níveis de certificação FIPS-140-2, FIPS-140-3 e FIDO. Você pode restringir o registro do autenticador nas políticas do IAM com base no tipo e no nível da certificação. Para obter mais informações, consulte as políticas abaixo.

Políticas de exemplo para certificações de dispositivos

Os seguintes casos de uso mostram exemplos de políticas que permitem registrar dispositivos MFA com certificações FIPS.

Tópicos

- Caso de uso 1: permitir o registro somente de dispositivos que tenham certificações FIPS-140-2 L2

- Caso de uso 2: permitir o registro de dispositivos que tenham certificações FIPS-140-2 L2 e FIDO L1
- Caso de uso 3: permitir o registro de dispositivos que tenham certificações FIPS-140-2 L2 ou FIPS-140-3 L2
- Caso de uso 4: Permitir o registro de dispositivos que tenham certificação FIPS-140-2 L2 e aceitem outros tipos de MFA, como autenticadores virtuais e TOTP de hardware

Caso de uso 1: permitir o registro somente de dispositivos que tenham certificações FIPS-140-2 L2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
}
```

Caso de uso 2: permitir o registro de dispositivos que tenham certificações FIPS-140-2 L2 e FIDO L1

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
}
```

Caso de uso 3: permitir o registro de dispositivos que tenham certificações FIPS-140-2 L2 ou FIPS-140-3 L2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
            "Effect": "Allow",
```

```
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
    ]
}
```

Caso de uso 4: Permitir o registro de dispositivos que tenham certificação FIPS-140-2 L2 e aceitem outros tipos de MFA, como autenticadores virtuais e TOTP de hardware

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Create"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:RegisterSecurityKey": "Activate",
          "iam:FIDO-FIPS-140-2-certification": "L2"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:EnableMFADevice",
      "Resource": "*",
      "Condition": {
        "Null": {
          "iam:RegisterSecurityKey": "true"
        }
      }
    }
  ]
}
```

AWS CLI e API da AWS

AWS A só é compatível com o uso de chaves de segurança no Console de gerenciamento da AWS. Não é possível usar de chaves de acesso e chaves de segurança para MFA na AWS CLI e na API da AWS, nem para acessar operações de API protegidas por MFA.

Recursos adicionais

- Para obter mais informações sobre como usar chaves de acesso e chaves de segurança na AWS, consulte Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS.
- Para obter ajuda com a solução de problemas de chaves de acesso e chaves de segurança na AWS, consulte Solucionar problemas de chaves de acesso e chaves de segurança FIDO.
- Para obter informações gerais do setor sobre compatibilidade com FIDO2, consulte FIDO2 Project.

Atribuir um dispositivo MFA virtual ao Console de gerenciamento da AWS

> **Important**
>
> A AWS recomenda usar uma chave de acesso ou chave de segurança para MFA na AWS sempre que possível. Para obter mais informações, consulte Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS.

É possível usar um telefone ou outro dispositivo como um dispositivo de autenticação multifator (MFA) virtual. Para fazer isso, instale um aplicativo móvel compatível com RFC 6238, um algoritmo TOTP (senha única baseada em tempo) baseado em padrões. Essas aplicações geram um código de autenticação de seis dígitos. Como os autenticadores podem ser executados em dispositivos móveis não protegidos e os códigos podem ser compartilhados com partes não autorizadas, a MFA baseada em TOTP não fornece o mesmo nível de segurança que o das opções de proteção contra phishing, como chaves de segurança e chaves de acesso FIDO2. Recomendamos usar chaves de acesso ou chaves de segurança para MFA para obter a proteção mais forte contra ataques como phishing.

Se ainda não for possível usar chaves de acesso ou chaves de segurança, recomendamos usar um dispositivo MFA virtual como medida provisória enquanto aguarda quaisquer aprovações de compra de hardware ou aguarda a chegada do hardware.

A maioria das aplicações de MFA virtual oferece suporte à criação de vários dispositivos virtuais, permitindo usar a mesma aplicação para várias Contas da AWS ou usuários. Você pode registrar até oito dispositivos com MFA de qualquer combinação dos tipos de MFA com seu Usuário raiz da conta da AWS e usuários do IAM. Basta um dispositivo MFA para acessar o Console de gerenciamento da AWS ou criar uma sessão pela AWS CLI. Recomendamos que você registre vários dispositivos

de MFA. Para aplicações autenticadoras, também recomendamos habilitar o recurso de backup ou de sincronização na nuvem para ajudar a evitar a perda de acesso à sua conta caso você perca ou quebre o dispositivo.

AWSA exige uma aplicação de MFA virtual que produz uma OTP de seis dígitos. Para obter uma lista de aplicativos de MFA virtual que você pode usar, consulte Autenticação multifator.

Tópicos

- Permissões obrigatórias
- Habilitar um dispositivo com MFA virtual para um usuário do IAM (console)
- Substituir um dispositivo de MFA virtual

Permissões obrigatórias

Para gerenciar dispositivos com MFA virtuais para o usuário do IAM, você deve ter as permissões na seguinte política: AWS: permite que os usuários do IAM autenticados por MFA gerenciem seus próprios dispositivos de MFA na página Credenciais de segurança.

Habilitar um dispositivo com MFA virtual para um usuário do IAM (console)

Você pode usar o IAM no Console de gerenciamento da AWS para habilitar e gerenciar um dispositivo com MFA virtual para um usuário do IAM em sua conta. Você pode associar tags aos seus recursos do IAM, incluindo dispositivos MFA virtuais, para identificar, organizar e controlar o acesso a eles. Os dispositivos MFA virtuais só podem ser marcados quando você usa a AWS CLI ou a API da AWS. Para habilitar e gerenciar um dispositivo MFA usando a AWS CLI ou a API da AWS, consulte Atribua dispositivos MFA na AWS CLI ou API da AWS. Para obter mais informações sobre recursos de marcação do IAM, consulte Tags para recursos do AWS Identity and Access Management.

> **Note**
>
> Você deve ter acesso físico ao hardware que hospedará o dispositivo MFA virtual do usuário para configurar a MFA. Por exemplo, você pode configurar a MFA para um usuário que usa um dispositivo MFA virtual executando em um smartphone. Neste caso, você precisa que o smartphone esteja disponível para concluir o assistente. Por isso, você pode optar por permitir que os usuários configurem e gerenciem seus próprios dispositivos MFA virtual. Neste caso, você deve conceder aos usuários as permissões para executar as ações necessárias do IAM. Para obter mais informações e ver um exemplo de política do IAM que

concede essas permissões, consulte Tutorial do IAM: Permitir que os usuários gerenciem suas credenciais e configurações de MFA e a política de exemplo AWS: permite que os usuários do IAM autenticados por MFA gerenciem seus próprios dispositivos de MFA na página Credenciais de segurança.

Para habilitar um dispositivo com MFA virtual para um usuário do IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Na lista Usuários, escolha o nome de usuário do IAM.
4. Selecione a guia Security Credentials (Credenciais de segurança). Em Multi-Factor Authentication (MFA) (autenticação multifator [MFA]), escolha Assign MFA device (Atribuir dispositivo de MFA).
5. No assistente, digite um Nome de dispositivo, escolha Aplicação de autenticador e escolha Próximo.

   O IAM gera e exibe informações de configuração para o dispositivo com MFA virtual, incluindo um código QR gráfico. O gráfico é uma representação da "chave de configuração secreta" que está disponível para entrada manual em dispositivos que não suportam códigos QR.
6. Abra o seu aplicativo de MFA virtual. Para obter uma lista de aplicativos que você pode usar para hospedar dispositivos MFA virtuais, consulte Autenticação multifator.

   Se o aplicativo de MFA virtual oferecer suporte a vários dispositivos ou contas de MFA virtual, selecione a opção para criar uma conta ou um dispositivo MFA virtual.
7. Determine se o aplicativo de MFA é compatível com códigos QR e, em seguida, execute uma das seguintes ações:

   - No assistente, escolha Mostrar código de QR e, em seguida, use o app para digitalizar o código de QR. Pode ser um ícone de câmera ou a opção Digitalizar código que usa a câmera do dispositivo para digitalizar o código.
   - No assistente , escolha Show secret key (Exibir chave secreta) e digite a chave secreta em sua aplicação de MFA.

   Quando você tiver concluído, o dispositivo MFA virtual inicia a geração de senhas de uso único.

8. Na página Configurar dispositivo, na caixa Código MFA 1, digite a senha de uso único que atualmente é exibida no dispositivo MFA virtual. Espere até 30 segundos para que o dispositivo gere uma nova senha de uso único. Em seguida, digite a segunda senha de uso único na caixa Código MFA 2. Escolha Add MFA (Adicionar MFA).

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA associa com êxito ao usuário, mas o dispositivo MFA está fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Caso isso ocorra, você pode ressincronizar o dispositivo.

O dispositivo MFA virtual está pronto para uso com a AWS. Para obter informações sobre como usar a MFA com o Console de gerenciamento da AWS, consulte Login habilitado para MFA.

> **Note**
>
> Os dispositivos virtuais de MFA não atribuídos na Conta da AWS são excluídos quando você adiciona novos dispositivos virtuais de MFA por meio do Console de gerenciamento da AWS ou durante o processo de login. Dispositivos virtuais de MFA não atribuídos são dispositivos na conta, mas que não são usados pelo usuário-raiz da conta ou pelos usuários do IAM no processo de login. Eles são excluídos para que novos dispositivos virtuais de MFA possam ser adicionados à conta. Também é possível reutilizar nomes de dispositivos.
>
> - Para visualizar dispositivos virtuais de MFA não atribuídos na conta, você pode usar o comando list-virtual-mfa-devices da AWS CLI ou a chamada de API.
> - Para desativar um dispositivo virtual de MFA, você pode usar o comando deactivate-mfa-device da AWS CLI ou a chamada de API. O dispositivo ficará sem atribuição.
> - Para anexar um dispositivo virtual de MFA não atribuído ao usuário-raiz da Conta da AWS ou aos usuários do IAM, você precisará do código de autenticação gerado pelo dispositivo junto com o comando enable-mfa-device da AWS CLI ou a chamada de API.

Substituir um dispositivo de MFA virtual

Seu Usuário raiz da conta da AWS e usuários do IAM podem registrar até oito dispositivos MFA de qualquer combinação de tipos de MFA. Se o usuário perder um dispositivo ou precisar substituí-lo por algum motivo, desative o dispositivo antigo. Em seguida, você pode adicionar o novo dispositivo para o usuário.

- Para desativar o dispositivo associado no momento a outro usuário do IAM, consulte Desativar um dispositivo com MFA.
- Para adicionar um dispositivo com MFA virtual de substituição para outro usuário do IAM, siga as etapas no procedimento Habilitar um dispositivo com MFA virtual para um usuário do IAM (console) acima.
- Para adicionar um dispositivo de MFA virtual de substituição para o Usuário raiz da conta da AWS, siga as etapas no procedimento Habilitar um dispositivo MFA virtual para o usuário-raiz (console).

Atribuir um token de hardware TOTP ao Console de gerenciamento da AWS

> **Important**
>
> A AWS recomenda usar uma chave de acesso ou chave de segurança para MFA na AWS sempre que possível. Para obter mais informações, consulte Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS.

Um token de hardware TOTP gera um código numérico de seis dígitos com base no algoritmo de senha de uso único com marcação temporal (TOTP). O usuário deve digitar um código válido no dispositivo quando solicitado durante o processo de login. Cada dispositivo MFA atribuído a um usuário deve ser exclusivo; um usuário não pode digitar um código no dispositivo de outro usuário para ser autenticado. Os dispositivos de MFA não podem ser compartilhados entre contas ou usuários.

Os tokens de hardware TOTP e chaves de segurança FIDO são dispositivos físicos que você compra. Os dispositivos físicos de MFA geram códigos de TOTP para autenticação quando você faz login na AWS. Eles dependem de baterias, que podem precisar ser substituídas e ressincronizadas com a AWS com o passar do tempo. As chaves de segurança FIDO, que utilizam criptografia de chave pública, não requerem baterias e oferecem um processo direto de autenticação. Recomendamos o uso de chaves de segurança FIDO por sua resistência a phishing, o que as

torna uma alternativa mais segura aos dispositivos de TOTP. Além disso, as chaves de segurança FIDO podem comportar vários usuários do IAM ou raízes no mesmo dispositivo, o que aumenta sua utilidade para proteção da conta. Para especificações e informações sobre aquisição para ambos os tipos de dispositivo, consulte Autenticação multifator.

Você pode habilitar um token de hardware TOTP para um usuário do IAM pelo Console de gerenciamento da AWS, pela linha de comando ou pela API do IAM. Para habilitar um dispositivo MFA para o Usuário raiz da conta da AWS, consulte Habilitar um token de hardware TOTP para o usuário-raiz (console).

Você pode registrar até oito dispositivos com MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu Usuário raiz da conta da AWS e usuários do IAM. Com vários dispositivos de MFA, basta um dispositivo de MFA para acessar o Console de gerenciamento da AWS ou criar uma sessão pela AWS CLI como esse usuário.

> **Important**
>
> Recomendamos habilitar vários dispositivos de MFA para que seus usuários tenham acesso contínuo à conta, caso um dispositivo de MFA seja perdido ou fique inacessível.

> **Note**
>
> Se quiser habilitar o dispositivo de MFA na linha de comando, use `aws iam enable-mfa-device`. Para habilitar o dispositivo com MFA com a API do IAM, use a operação `EnableMFADevice`.

Tópicos

- Permissões obrigatórias
- Habilitar um token de hardware TOTP para seu próprio usuário do IAM (console)
- Habilitar um token de hardware TOTP para outro usuário do IAM (console)
- Substituir um dispositivo de MFA físico

Permissões obrigatórias

Para gerenciar um token de hardware TOTP para seu próprio usuário do IAM ao proteger ações confidenciais relacionadas a MFA, você deve ter as permissões na seguinte política:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Habilitar um token de hardware TOTP para seu próprio usuário do IAM (console)

Você pode habilitar seu próprio token de hardware TOTP no Console de gerenciamento da AWS.

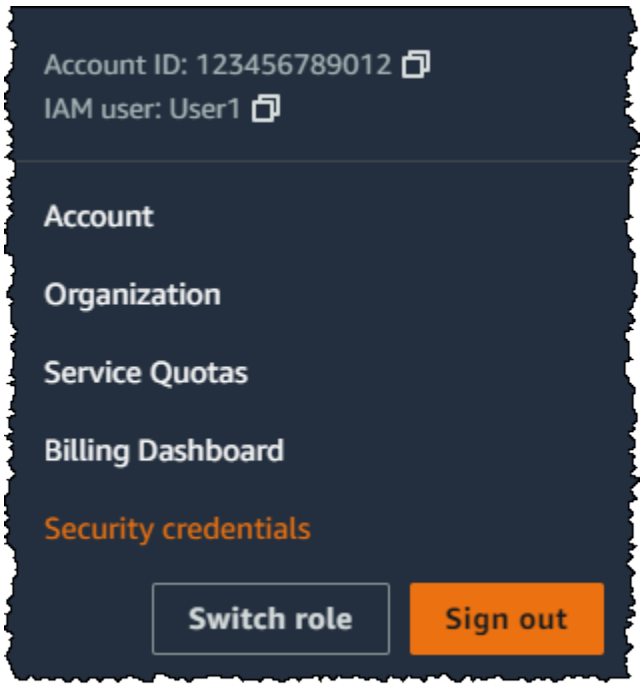
> **Note**
>
> Para poder habilitar um token de hardware TOTP, é necessário ter acesso físico ao dispositivo.

Para habilitar um token de hardware TOTP para seu próprio usuário do IAM (console)

1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

   > **Note**
   >
   > Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

   Para obter o ID da Conta da AWS, fale com o administrador.
2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na guia Credenciais do AWS IAM, na seção Autenticação multifator (MFA), escolha Atribuir dispositivo com MFA.
4. No assistente, digite um Device name (Nome de dispositivo), escolha Hardware TOTP token (Token de hardware TOTP) e escolha Next (Avançar).
5. Digite o número de série do dispositivo. O número de série é geralmente encontrado na parte de trás do dispositivo.
6. Na caixa MFA code 1 (Código MFA 1), digite o número de seis dígitos exibido pelo dispositivo MFA. Talvez seja necessário pressionar o botão na parte frontal do dispositivo para exibir o número.

7. Aguarde 30 segundos enquanto o dispositivo atualiza o código e digite o próximo número de seis dígitos na caixa MFA code 2 (Código MFA 2). Talvez seja necessário pressionar o botão na parte frontal do dispositivo novamente para exibir o segundo número.
8. Escolha Add MFA (Adicionar MFA).

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos de autenticação. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA será

associado com êxito ao usuário, mas o dispositivo MFA ficará fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Caso isso ocorra, você pode ressincronizar o dispositivo.

O dispositivo está pronto para uso com a AWS. Para obter informações sobre como usar a MFA com o Console de gerenciamento da AWS, consulte Login habilitado para MFA.

Habilitar um token de hardware TOTP para outro usuário do IAM (console)

Você pode habilitar um token de hardware TOTP para outro usuário do IAM no Console de gerenciamento da AWS.

Para habilitar um token de hardware TOTP para outro usuário do IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Selecione o nome do usuário para o qual deseja habilitar a MFA.
4. Selecione a guia Security Credentials (Credenciais de segurança). Em Multi-Factor Authentication (MFA) (autenticação multifator [MFA]), escolha Assign MFA device (Atribuir dispositivo de MFA).
5. No assistente, digite um Device name (Nome de dispositivo), escolha Hardware TOTP token (Token de hardware TOTP) e escolha Next (Avançar).
6. Digite o número de série do dispositivo. O número de série é geralmente encontrado na parte de trás do dispositivo.
7. Na caixa MFA code 1 (Código MFA 1), digite o número de seis dígitos exibido pelo dispositivo MFA. Talvez seja necessário pressionar o botão na parte frontal do dispositivo para exibir o número.

8. Aguarde 30 segundos enquanto o dispositivo atualiza o código e digite o próximo número de seis dígitos na caixa MFA code 2 (Código MFA 2). Talvez seja necessário pressionar o botão na parte frontal do dispositivo novamente para exibir o segundo número.

9. Escolha Add MFA (Adicionar MFA).

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos de autenticação. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA será associado com êxito ao usuário, mas o dispositivo MFA ficará fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Caso isso ocorra, você pode ressincronizar o dispositivo.

O dispositivo está pronto para uso com a AWS. Para obter informações sobre como usar a MFA com o Console de gerenciamento da AWS, consulte Login habilitado para MFA.

Substituir um dispositivo de MFA físico

Você pode ter até oito dispositivos com MFA de qualquer combinação dos tipos de MFA atualmente compatíveis atribuídos a um usuário ao mesmo tempo com seu Usuário raiz da conta da AWS e usuários do IAM. Se o usuário perde um dispositivo ou precisa substituí-lo por algum motivo, você deve primeiro desativar o dispositivo antigo. Em seguida, você pode adicionar o novo dispositivo para o usuário.

- Para desativar o dispositivo atualmente associado a um usuário, consulte Desativar um dispositivo com MFA.
- Para adicionar um token de hardware TOTP de substituição para um usuário do IAM, siga as etapas do procedimento Habilitar um token de hardware TOTP para outro usuário do IAM (console) anterior deste tópico.
- Para adicionar um token de hardware TOTP de substituição para o Usuário raiz da conta da AWS, siga as etapas do procedimento Habilitar um token de hardware TOTP para o usuário-raiz (console) anterior neste tópico.

Atribua dispositivos MFA na AWS CLI ou API da AWS

Você pode usar os comandos da AWS CLI ou operações de API da AWS para habilitar um dispositivo com MFA virtual para um usuário do IAM. Não é possível habilitar um dispositivo com MFA para o Usuário raiz da conta da AWS com a AWS CLI, a API da AWS, ferramentas para

Windows PowerShell ou qualquer outra ferramenta de linha de comando. No entanto, você pode usar o Console de gerenciamento da AWS para habilitar um dispositivo com MFA para o usuário raiz.

Quando você habilita um dispositivo MFA do Console de gerenciamento da AWS, o console executa múltiplas etapas para você. Se, em vez disso, você criar um dispositivo virtual usando a AWS CLI, o Tools for Windows PowerShell ou a API da AWS, execute as etapas manualmente e na ordem correta. Por exemplo, para criar um dispositivo com MFA virtual, você deve criar o objeto do IAM e extrair o código como uma string ou um gráfico de código QR. Em seguida, você deve sincronizar o dispositivo e associá-lo a um usuário do IAM. Consulte a seção Exemplos do New-IAMVirtualMFADevice para obter mais detalhes. Para um dispositivo físico, ignore a etapa de criação, sincronize e associe o dispositivo ao usuário diretamente.

Você pode associar tags aos seus recursos do IAM, incluindo dispositivos MFA virtuais, para identificar, organizar e controlar o acesso a eles. Os dispositivos MFA virtuais só podem ser marcados quando você usa a AWS CLI ou a API da AWS.

Um usuário do IAM usando o SDK ou a CLI pode ativar um dispositivo de MFA adicional chamando `EnableMFADevice` ou desativar um dispositivo de MFA existente por meio de uma chamada `DeactivateMFADevice`. Para fazer isso com êxito, eles devem primeiro chamar `GetSessionToken` e enviar códigos de MFA com um dispositivo de MFA existente. Essa chamada retorna credenciais de segurança temporárias que podem ser usadas para assinar operações de API que exigem autenticação de MFA. Para ver um exemplo de solicitação e resposta, consulte `GetSessionToken`: credenciais temporárias para usuários em ambientes não confiáveis.

Para criar a entidade do dispositivo virtual no IAM para representar um dispositivo virtual de MFA

Esses comandos fornecem um ARN para o dispositivo que é usado no lugar de um número de série em muitos dos comandos a seguir.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API: `CreateVirtualMFADevice`

Para habilitar um dispositivo com MFA para usar com a AWS

Esses comandos sincronizam o dispositivo com a AWS e associam-no a um usuário. Se o dispositivo for virtual, use o ARN do dispositivo virtual como número de série.

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos de autenticação. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA será associado com êxito ao usuário, mas o dispositivo MFA ficará fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período. Se isso acontecer, sincronize novamente o dispositivo usando os comandos descritos abaixo.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API: `EnableMFADevice`

Para desativar um dispositivo

Use estes comandos para desassociar o dispositivo do usuário e desativá-lo. Se o dispositivo for virtual, use o ARN do dispositivo virtual como número de série. Também é necessário excluir, separadamente, a entidade do dispositivo virtual.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API: `DeactivateMFADevice`

Para listar entidades do dispositivo virtual de MFA

Use estes comandos para listar entidades de dispositivo MFA virtual.

- AWS CLI: `aws iam list-virtual-mfa-devices`
- AWS API: `ListVirtualMFADevices`

Marcar um dispositivo MFA virtual

Use estes comandos para marcar um dispositivo MFA virtual.

- AWS CLI: `aws iam tag-mfa-device`
- AWS API: `TagMFADevice`

Listar tags para um dispositivo MFA virtual

Use estes comandos para listar as tags associadas a um dispositivo MFA virtual.

- AWS CLI: `aws iam list-mfa-device-tags`
- AWS API: `ListMFADeviceTags`

Desmarcar um dispositivo MFA virtual

Use estes comandos para remover as tags associadas a um dispositivo MFA virtual.

- AWS CLI: `aws iam untag-mfa-device`
- AWS API: `UntagMFADevice`

Para sincronizar novamente um dispositivo MFA

Use estes comandos se o dispositivo estiver gerando códigos que não são aceitos pela AWS. Se o dispositivo for virtual, use o ARN do dispositivo virtual como número de série.

- AWS CLI: `aws iam resync-mfa-device`
- AWS API: `ResyncMFADevice`

Para excluir uma entidade do dispositivo com MFA virtual no IAM

Após o dispositivo ser desassociado do usuário, você poderá excluir a entidade do dispositivo.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API: `DeleteVirtualMFADevice`

Para recuperar um dispositivo MFA virtual que foi perdido ou não está funcionando

Às vezes, o dispositivo de um usuário que hospeda a aplicação de MFA virtual é perdido, substituído ou não está funcionando. Quando isso acontece, o usuário não pode recuperá-lo sozinho. Os usuários devem entrar em contato com o administrador para desativar o dispositivo. Para obter mais informações, consulte Recuperar uma identidade protegida por MFA no IAM.

Verificar status da MFA

Use o console do IAM para verificar se um Usuário raiz da conta da AWS ou usuário do IAM tem um dispositivo com MFA válido habilitado.

Para verificar o status de MFA de um usuário raiz

1. Faça login no Console de gerenciamento da AWS com suas credenciais de usuário raiz e, em seguida, abra o console do IAM no https://console.aws.amazon.com/iam/.
2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).
3. Consulte Multi-factor Authentication (MFA) (Autenticação multifator, MFA) para ver se o MFA está habilitado ou desabilitado. Se a MFA não tiver sido ativada, um símbolo de alerta ( ) será exibido.

Se você quiser habilitar a MFA para a conta, consulte um dos seguintes:

- Habilitar um dispositivo MFA virtual para o usuário-raiz (console)
- Habilitar uma chave de acesso ou uma chave de segurança para o usuário-raiz (console)
- Habilitar um token de hardware TOTP para o usuário-raiz (console)

Para verificar o status de MFA de usuários do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Users.
3. Se necessário, adicione a coluna MFA à tabela de usuários concluindo as etapas a seguir:
   a. Acima da tabela, no canto direito, selecione o ícone de configurações ( ).
   b. Em Gerenciar colunas, selecione MFA.
   c. (Opcional) Desmarque a caixa de seleção para qualquer cabeçalho de coluna que você não deseje exibir na tabela de usuários.
   d. Escolha Fechar para retornar à lista de usuários.
4. A coluna MFA fornece informações sobre o dispositivo MFA que está habilitado. Se não houver um dispositivo MFA ativo para o usuário, o console exibirá None (Nenhum). Se o usuário tiver um dispositivo de MFA habilitado, a coluna MFA exibirá o tipo de dispositivo que está habilitado com um valor Virtual, Security Key (Chave de segurança), Hardware ou SMS.

> **Note**
>
> A AWS não é mais compatível com a habilitação de autenticação multifator (MFA) por SMS. Recomendamos que os clientes que têm usuários de IAM que usam MFA baseada em texto de SMS mudem para um dos seguintes métodos alternativos: dispositivo de MFA virtual (baseado em software), chave de segurança FIDO ou dispositivo de MFA de hardware. Você pode identificar os usuários da sua conta com um dispositivo de MFA por SMS atribuído. Para fazer isso, vá para o console do IAM, escolha Users (Usuários) no painel de navegação e procure os usuários com SMS na coluna MFA da tabela.

5. Para visualizar informações adicionais sobre o dispositivo MFA para um usuário, escolha o nome do usuário cujo status de MFA você deseja verificar. Em seguida, selecione a guia Credenciais de segurança.
6. Se não houver um dispositivo com MFA ativo para o usuário, o console exibirá Nenhum dispositivo com MFA. Atribua um dispositivo com MFA para melhorar a segurança de seu ambiente da AWS na seção Autenticação multifator (MFA). Se o usuário tiver dispositivos de MFA habilitados, a seção Multi-factor authentication (MFA) (Autenticação multifator [MFA]) exibirá detalhes sobre os dispositivos:
   - O nome do dispositivo
   - O tipo do dispositivo
   - O identificador do dispositivo, como o número de série de um dispositivo físico ou o ARN de um dispositivo virtual na AWS
   - Quando o dispositivo foi criado

Para remover ou ressincronizar um dispositivo, escolha o botão de opção ao lado do dispositivo e escolha Remove (Remover) ou Resync (Ressincronizar).

Para obter mais informações sobre como habilitar a MFA, consulte o seguinte:

- Atribuir um dispositivo MFA virtual ao Console de gerenciamento da AWS
- Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS
- Atribuir um token de hardware TOTP ao Console de gerenciamento da AWS

Sincronizar novamente os dispositivos MFA virtuais e de hardware

Você pode usar a AWS para sincronizar novamente seus dispositivos de autenticação multifator (MFA) virtuais e de hardware. Se o seu dispositivo não estiver sincronizado quando você tentar usá-lo, ocorrerá uma falha na tentativa de login, e o IAM solicitará que você sincronize o dispositivo novamente.

> **Note**
>
> As chaves de segurança FIDO não perdem a sincronia. Se uma chave de segurança FIDO for perdida ou rompida, você poderá desativá-la. Para obter instruções sobre a desativação de qualquer tipo de dispositivo MFA, consulte Para desativar um dispositivo com MFA para outro usuário do IAM (console).

Como administrador da AWS, você pode sincronizar novamente os dispositivos com MFA virtuais e de hardware dos usuários do IAM se eles perderem a sincronização.

Se o seu dispositivo com MFA de Usuário raiz da conta da AWS não estiver funcionando, você poderá sincronizar novamente o dispositivo usando o console do IAM, concluindo ou não o processo de login. Se você não conseguir ressincronizar seu dispositivo, talvez seja necessário desassociá-lo e reassociá-lo. Para obter mais informações sobre como fazer isso, consulte Desativar um dispositivo com MFA e AWSCódigo da autenticação multifator no IAM da.

Tópicos

- Permissões obrigatórias
- Sincronizar novamente dispositivos com MFA virtuais e de hardware (console do IAM)
- Sincronizar novamente dispositivos com MFA virtuais e de hardware (AWS CLI)
- Sincronizar novamente dispositivos com MFA virtuais e de hardware (API da AWS)

Permissões obrigatórias

Para dessincronizar dispositivos com MFA virtuais ou de hardware para o usuário do IAM, você deve ter as permissões desta política. Esta política não permite que você crie ou desative um dispositivo.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Sincronizar novamente dispositivos com MFA virtuais e de hardware (console do IAM)

Você pode usar o console do IAM para sincronizar novamente dispositivos com MFA virtuais e de hardware.

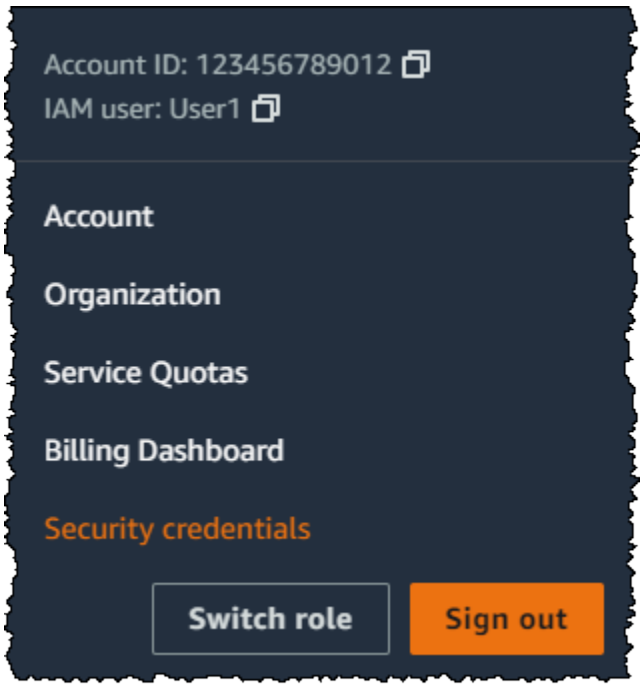
Para sincronizar novamente um dispositivo com MFA virtual ou de hardware para seu próprio usuário do IAM (console)

1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

   > **Note**
   >
   > Para sua conveniência, a página de login da AWS usa um cookie do navegador para lembrar seu nome de usuário e as informações da conta do IAM. Se você já tiver feito login como outro usuário, escolha Sign in to a different account (Fazer login com uma conta diferente) próximo à parte inferior da página para retornar à página de login principal. Daí, você pode inserir o ID ou o alias da conta da AWS para ser redirecionado para a página de login de usuário do IAM da sua conta.

   Para obter o ID da Conta da AWS, fale com o administrador.

2. No canto superior direito da barra de navegação, escolha seu nome de usuário e selecione Security credentials (Credenciais de segurança).



3. Na guia Credenciais do AWS IAM, na seção Autenticação multifator (MFA), escolha o botão de opção ao lado do dispositivo com MFA e escolha Sincronizar novamente.
4. Digite os próximos dois códigos gerados sequencialmente no dispositivo em MFA code 1 (Código MFA 1) e MFA code 2 (Código MFA 2). Em seguida, escolha Resync (Ressincronizar).

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, a solicitação parecerá funcionar, mas o dispositivo permanecerá fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período.

Para sincronizar novamente um dispositivo com MFA virtual ou de hardware para outro usuário do IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Usuários e, em seguida, escolha o nome do usuário cujo dispositivo MFA precisa ser sincronizado novamente.

3. Selecione a guia Credenciais de segurança. Na seção Autenticação multifator (MFA), escolha o botão de opção ao lado do dispositivo com MFA e escolha Sincronizar novamente.
4. Digite os próximos dois códigos gerados sequencialmente no dispositivo em MFA code 1 (Código MFA 1) e MFA code 2 (Código MFA 2). Em seguida, escolha Resync (Ressincronizar).

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, a solicitação parecerá funcionar, mas o dispositivo permanecerá fora de sincronia. Isso ocorre porque as senhas únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período.

Para sincronizar novamente sua MFA de usuário raiz antes de fazer login (console)

1. Na página Amazon Web Services Sign In With Authentication Device (Login da Amazon Web Services com dispositivo de autenticação), escolha Having problems with your authentication device? (Está com problemas com seu dispositivo de autenticação?) Clique aqui.

> **Note**
>
> É possível ver um texto diferente, como Fazer login usando MFA e Solucionar problemas do dispositivo de autenticação. No entanto, os mesmos recursos são fornecidos.

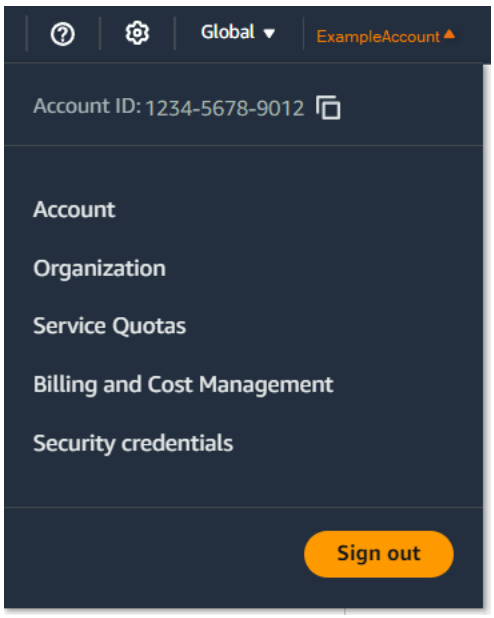
2. Na seção Re-Sync With Our Servers (Sincronizar novamente com nossos servidores), digite os próximos dois códigos gerados sequencialmente no dispositivo em MFA code 1 (Código MFA 1) e MFA code 2 (Código MFA 2). Em seguida, escolha Sincronizar novamente dispositivo de autenticação.
3. Se necessário, digite sua senha novamente e escolha Faça login. Em seguida, conclua o login usando seu dispositivo MFA.

Para sincronizar novamente seu dispositivo com MFA de usuário raiz depois de fazer login (console)

1. Faça login no console do IAM como proprietário da conta escolhendo Root user (Usuário root) e inserindo o endereço de e-mail da sua Conta da AWS. Na próxima página, insira sua senha.

> **Note**
>
> Como usuário raiz, você não pode acessar a página Fazer login como usuário do IAM. Ao visualizar a página de login de usuário do IAM, escolha a opção Fazer login usando o e-mail de usuário raiz, próximo à parte inferior da página. Para obter ajuda para fazer login como usuário raiz, consulte Fazer login no Console de gerenciamento da AWS como usuário raiz no Guia do usuário do Início de Sessão da AWS.

2. No lado direito da barra de navegação, selecione seu nome de conta e selecione Credenciais de segurança. Se necessário, selecione Continue to Security credentials (Prosseguir para as credenciais de segurança).



3. Expanda a seção Multi-factor authentication (MFA) (Autenticação multifator (MFA)) na página.
4. Selecione o botão de opção ao lado do dispositivo e escolha Resync (Ressincronizar).
5. Na caixa de diálogo Resync MFA device (Ressincronizar dispositivo de MFA), digite os próximos dois códigos gerados sequencialmente no dispositivo em MFA code 1 (Código MFA 1) e MFA code 2 (Código MFA 2). Em seguida, escolha Resync (Ressincronizar).

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, o dispositivo MFA se associa com êxito ao usuário, mas o dispositivo MFA está fora de sincronia. Isso ocorre porque as senhas

> únicas baseadas em tempo (time-based one-time passwords, TOTP) expiram após um curto período.

Sincronizar novamente dispositivos com MFA virtuais e de hardware (AWS CLI)

Você pode sincronizar novamente os dispositivos MFA virtuais e de hardware da AWS CLI.

Para sincronizar novamente um dispositivo com MFA virtual ou de hardware para um usuário do IAM (AWS CLI)

Em um prompt de comando, emita o comando aws iam resync-mfa-device:

- Dispositivo MFA virtual: especifique o nome de recurso da Amazon (ARN) do dispositivo como o número de série.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Dispositivo MFA de hardware: especifique o número de série do dispositivo de hardware como o número de série. O formato é específico do fornecedor. Por exemplo, você pode comprar um token gemalto da Amazon. Seu número de série geralmente tem quatro letras seguidas por quatro números.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Envie sua solicitação imediatamente após gerar os códigos. Se você gerar os códigos e esperar muito tempo para enviar a solicitação, ocorrerá falha na solicitação porque os códigos expiram após um curto período.

Sincronizar novamente dispositivos com MFA virtuais e de hardware (API da AWS)

O IAM tem uma chamada de API que executa a sincronização. Nesse caso, recomendamos que você atribua aos seus usuários de dispositivos MFA virtuais e de hardware permissão para acessar

essa chamada de API. Em seguida, crie uma ferramenta com base na chamada de API que permite que seus usuários sincronizem novamente seus dispositivos sempre que precisarem.

Para sincronizar novamente um dispositivo com MFA virtual ou de hardware para um usuário do IAM (API da AWS)

- Envie a solicitação ResyncMFADevice.

Desativar um dispositivo com MFA

Caso enfrente problemas para fazer login com um dispositivo de com autenticação multifator (MFA) como um usuário do IAM, entre em contato com o administrador para obter ajuda.

Como administrador, você pode desativar o dispositivo para outro usuário do IAM. Isso permite que o usuário conecte-se sem usar o MFA. Você pode fazer isso como uma solução temporária enquanto o dispositivo MFA é substituído, ou se o dispositivo está temporariamente indisponível. No entanto, recomendamos que você ative um novo dispositivo para o usuário assim que possível. Para saber como ativar um novo dispositivo MFA, consulte AWSCódigo da autenticação multifator no IAM da.

> **Note**
>
> Se você usar a API ou a AWS CLI para excluir um usuário da sua Conta da AWS, desative ou exclua o dispositivo de MFA do usuário. Você faz essa alteração como parte do processo de remoção do usuário. Para obter mais informações sobre a remoção de usuários, consulte Remover ou desativar um usuário do IAM.

Tópicos

- Desativar dispositivos MFA (console)
- Desativar dispositivos MFA (AWS CLI)
- Desativar dispositivos MFA (API da AWS)

Desativar dispositivos MFA (console)

Para desativar um dispositivo com MFA para outro usuário do IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação, escolha Usuários.
3. Para desativar o dispositivo MFA para um usuário, escolha o nome do usuário cuja MFA que você deseja remover.
4. Selecione a guia Credenciais de segurança.
5. Em Autenticação multifator (MFA), escolha o botão de opção ao lado do dispositivo com MFA, escolha Remover e depois Remover.

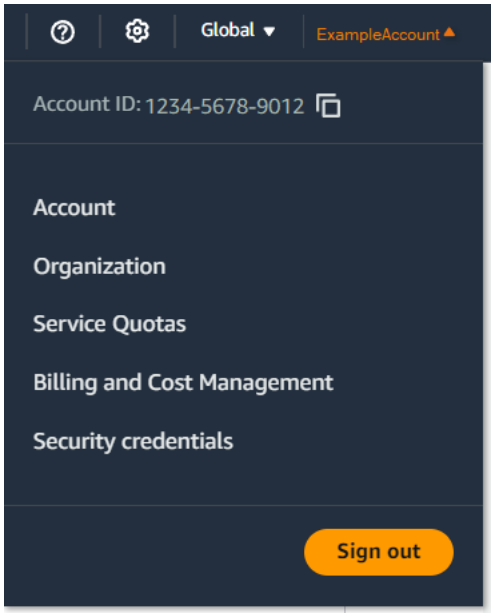
   O dispositivo é removido da AWS. Ele não poderá ser usado para fazer login nem autenticar as solicitações até ser reativado e associado a um usuário da AWS ou ao Usuário raiz da conta da AWS.

Para desativar o dispositivo MFA para seu Usuário raiz da conta da AWS (console)

1. Faça login no console do IAM como proprietário da conta escolhendo Root user (Usuário root) e inserindo o endereço de e-mail da sua Conta da AWS. Na próxima página, insira sua senha.

   > **Note**
   >
   > Como usuário raiz, você não pode acessar a página Fazer login como usuário do IAM. Ao visualizar a página de login de usuário do IAM, escolha a opção Fazer login usando o e-mail de usuário raiz, próximo à parte inferior da página. Para obter ajuda para fazer login como usuário raiz, consulte Fazer login no Console de gerenciamento da AWS como usuário raiz no Guia do usuário do Início de Sessão da AWS.

2. No lado direito da barra de navegação, selecione seu nome de conta e selecione Credenciais de segurança. Se necessário, selecione Continue to Security credentials (Prosseguir para as credenciais de segurança).



3. Na seção Multi-factor authentication (MFA) (Autenticação multifator [MFA]), escolha o botão de opção ao lado do dispositivo de MFA que você deseja desativar e escolha Remove (Remover).
4. Escolha Remover.

   O dispositivo de MFA é desativado para a Conta da AWS. Verifique o e-mail associado à sua Conta da AWS para ver se há uma mensagem de confirmação da Amazon Web Services. O e-mail informa que sua autenticação multifator (MFA) da Amazon Web Services foi desativada. A mensagem virá de `@amazon.com` ou `@aws.amazon.com`.

> **Note**
>
> Os dispositivos virtuais de MFA não atribuídos na Conta da AWS são excluídos quando você adiciona novos dispositivos virtuais de MFA por meio do Console de gerenciamento da AWS ou durante o processo de login. Dispositivos virtuais de MFA não atribuídos são dispositivos na conta, mas que não são usados pelo usuário-raiz da conta ou pelos usuários do IAM no processo de login. Eles são excluídos para que novos dispositivos virtuais de MFA possam ser adicionados à conta. Também é possível reutilizar nomes de dispositivos.

Desativar dispositivos MFA (AWS CLI)

Para desativar um dispositivo com MFA para um usuário do IAM (AWS CLI)

- Execute este comando: `aws iam deactivate-mfa-device`

Desativar dispositivos MFA (API da AWS)

Para desativar um dispositivo com MFA para um usuário do IAM (API da AWS)

- Chame esta operação: `DeactivateMFADevice`

Recuperar uma identidade protegida por MFA no IAM

Se o dispositivo de MFA virtual ou se o token de hardware TOTP estiver funcionando corretamente, mas você não conseguir usá-lo para acessar seus recursos da AWS, talvez ele não esteja sincronizado com a AWS. Para obter informações sobre a sincronização de um dispositivo MFA virtual ou de um dispositivo MFA de hardware, consulte Sincronizar novamente os dispositivos MFA virtuais e de hardware. As chaves de segurança FIDO não perdem a sincronia.

Se o dispositivo MFA para um Usuário raiz da conta da AWS for perdido, danificado ou não funcionar, você poderá recuperar o acesso à sua conta. Os usuários do IAM devem entrar em contato com um administrador para desativar o dispositivo.

> **Important**
>
> Recomendamos que você ative vários dispositivos MFA. O registro de vários dispositivos MFA ajudará a garantir o acesso contínuo se um dispositivo for perdido ou quebrado. Seu Usuário raiz da conta da AWS e usuários do IAM podem registrar até oito dispositivos MFA de qualquer tipo.

Pré-requisito: usar outro dispositivo de MFA

Se seu dispositivo de autenticação multifator (MFA) for perdido, danificado ou não estiver funcionando, você poderá fazer login usando outro dispositivo de MFA registrado no mesmo usuário raiz ou usuário do IAM.

Para iniciar sessão usando outro dispositivo de MFA

1. Inicie sessão no Console de gerenciamento da AWS com o ID da sua Conta da AWS ou alias e senha da conta.
2. Na página Verificação adicional necessária ou na página Autenticação multifator, escolha Tentar outro método de MFA.
3. Faça a autenticação com o tipo de dispositivo de MFA que você selecionou.

4. A próxima etapa será diferente dependendo de você ter conseguido iniciar sessão com um dispositivo de MFA alternativo.
   - Se você iniciou sessão com sucesso, poderá Sincronizar novamente os dispositivos MFA virtuais e de hardware, o que talvez resolva o problema. Se seu dispositivo de MFA for perdido ou quebrar, você poderá desativá-lo. Para obter instruções sobre a desativação de qualquer tipo de dispositivo MFA, consulte Desativar um dispositivo com MFA.
   - Se você não conseguir iniciar sessão com a MFA, use as etapas Recuperar um dispositivo com MFA de usuário raiz ou Recuperar um dispositivo com MFA de usuário do IAM para recuperar sua identidade protegida por MFA.

Recuperar um dispositivo com MFA de usuário raiz

Se você não conseguir iniciar sessão com o seu dispositivo de MFA, poderá usar métodos de autenticação alternativos comprovando sua identidade usando o e-mail e o telefone registrados na sua conta.

Confirme se você consegue acessar o e-mail e o número de telefone de contato principal associados à sua conta antes de usar fatores de autenticação alternativos para fazer login como usuário-raiz. Se você precisar atualizar o telefone de contato principal, faça login como usuário do IAM com acesso de Administrador em vez de usuário-raiz. Para obter instruções adicionais sobre como atualizar as informações de contato da conta, consulte Editar informações de contato no Guia do usuário do AWS Billing. Caso não tenha acesso a um e-mail e telefone de contato principal, fale com o AWS Support.

> **Important**
>
> Recomendamos manter atualizados o endereço de e-mail e o telefone de contato vinculados a seu usuário raiz para ter uma recuperação da conta bem-sucedida. Para obter mais informações, consulte Atualizar seu contato principal na sua Conta da AWS no Guia de Referência do AWS Gerenciamento de contas.

Para fazer login usando fatores alternativos de autenticação como um Usuário raiz da conta da AWS

1. Faça login no Console de gerenciamento da AWS como o proprietário da conta ao escolher a opção Usuário-raiz e inserir o endereço de e-mail da Conta da AWS. Na próxima página, insira a senha.

2. Na página Verificação adicional necessária, selecione um método de MFA para autenticação e escolha Avançar.

> **Note**
>
> Você pode ver um texto alternativo, como Sign in using MFA (Fazer login usando MFA), Troubleshoot your authentication device (Solucione o problema do dispositivo de autenticação) ou Troubleshoot MFA (Solucione o problema de MFA), mas a funcionalidade é a mesma. Se você não puder usar fatores de autenticação alternativos para verificar o endereço de e-mail e o número de telefone primário da conta, entre em contato com o AWS Support para desativar seu dispositivo de MFA.

3. Dependendo do tipo de MFA que estiver usando, você verá uma página diferente, mas a opção Solucionar problemas de MFA funciona da mesma forma. Na página Verificação adicional necessária ou na página Autenticação multifator, escolha Solucionar problemas de MFA.
4. Se necessário, digite sua senha novamente e escolha Conectar.
5. Na página Solucionar problemas do dispositivo de autenticação, na seção Acesso usando fatores alternativos de autenticação, escolha Acesse usando fatores alternativos.
6. Na página Entrar usando fatores alternativos de autenticação, autentique sua conta verificando o endereço de e-mail e escolha Enviar e-mail de verificação.
7. Verifique o e-mail associado à sua Conta da AWS para ver se há uma mensagem da Amazon Web Services (no-reply-aws@amazon.com). Siga as orientações no e-mail.

   Se você não vir o e-mail em sua conta, verifique sua pasta de spam ou retorne para seu navegador e escolha Reenviar o e-mail.
8. Depois de verificar seu endereço de e-mail, você pode continuar a autenticar sua conta. Para verificar seu número de telefone para contato primário, escolha Call me now (Ligar para mim agora).
9. Atenda a ligação da AWS e, quando receber a solicitação, digite o número de 6 dígitos do site AWS no teclado do seu telefone.

   Se você não receber uma chamada de AWS, escolha Fazer login para entrar no console novamente e começar novamente. Ou consulte Lost or unusable Multi-Factor Authentication (MFA) device (Dispositivo de autenticação multifator (MFA) perdido ou inutilizado para entrar em contato com o suporte e obter ajuda.

10. Depois de verificar o seu número de telefone, você pode fazer login na sua conta, escolhendo Fazer login no console.
11. A próxima etapa varia dependendo do tipo de MFA que você está usando:

    - Para um dispositivo MFA virtual, remova a conta do seu dispositivo. Em seguida, vá para a página Credenciais de segurança da AWS e exclua a entidade de dispositivo virtual de MFA antiga antes de criar uma nova.
    - Para obter uma chave de segurança FIDO, consulte a página AWS Security Credentials e desative a chave de segurança FIDO antiga antes de habilitar uma nova.
    - Para um token de hardware TOTP, fale com o provedor de terceiros para obter ajuda para corrigir ou substituir o dispositivo. Você pode continuar a fazer login usando fatores alternativos de autenticação até que você receba o novo dispositivo. Depois que tiver o novo dispositivo MFA de hardware, consulte a página AWS Security Credentials e exclua o dispositivo MFA antigo.

> **Note**
>
> Você não precisa substituir um dispositivo MFA perdido ou roubado pelo mesmo tipo de dispositivo. Por exemplo, se você quebrar a chave de segurança FIDO e solicitar uma nova, poderá usar um dispositivo MFA virtual ou um token de hardware TOTP até receber uma nova chave de segurança FIDO.

> **Important**
>
> Se seu dispositivo MFA estiver perdido ou tiver sido roubado, altere a senha de usuário-raiz depois de iniciar sessão e estabelecer seu dispositivo MFA substituto. Um invasor pode ter roubado o dispositivo de autenticação e também pode ter sua senha atual. Para obter mais informações, consulte Alterar a senha para o Usuário raiz da conta da AWS.

Recuperar um dispositivo com MFA de usuário do IAM

Se você for um usuário do IAM que não consegue iniciar sessão com MFA, não poderá recuperar um dispositivo de MFA por conta própria. Entre em contato com o administrador para desativar o dispositivo. Depois você poderá habilitar um novo dispositivo.

Para obter ajuda para um dispositivo MFA associado a um usuário do IAM

1. Entre em contato com o administrador da AWS ou outra pessoa que forneceu a você o nome de usuário e a senha para o usuário do IAM. O administrador deve desativar o dispositivo MFA, como descrito em Desativar um dispositivo com MFA para que você possa se conectar.
2. A próxima etapa varia dependendo do tipo de MFA que você está usando:

   - Para um dispositivo MFA virtual, remova a conta do seu dispositivo. Em seguida, ative o dispositivo virtual, como descrito em Atribuir um dispositivo MFA virtual ao Console de gerenciamento da AWS.
   - Para uma chave de segurança FIDO, entre em contato com o provedor terceirizado para obter ajuda para substituir o dispositivo. Quando você receber a nova chave de segurança FIDO, habilite-a, como descrito em Atribuir uma chave de acesso ou uma chave de segurança ao Console de gerenciamento da AWS.
   - Para um token de hardware TOTP, fale com o provedor de terceiros para obter ajuda para corrigir ou substituir o dispositivo. Após ter o novo dispositivo MFA físico, ative o dispositivo, como descrito em Atribuir um token de hardware TOTP ao Console de gerenciamento da AWS.

   > **Note**
   >
   > Você não precisa substituir um dispositivo MFA perdido ou roubado pelo mesmo tipo de dispositivo. É possível ter até oito dispositivos de MFA de qualquer combinação. Por exemplo, se você quebrar a chave de segurança FIDO e solicitar uma nova, poderá usar um dispositivo MFA virtual ou um token de hardware TOTP até receber uma nova chave de segurança FIDO.

3. Se seu dispositivo MFA foi perdido ou roubado, também altere sua senha no caso de um invasor ter roubado o dispositivo de autenticação e tenha também sua senha atual. Para obter mais informações, consulte Gerenciar senhas para usuários do IAM

Acesso seguro à API com a MFA

Com as políticas do IAM, você pode especificar quais operações de API um usuário tem permissão para chamar. Você pode adicionar mais segurança exigindo que os usuários se autentiquem

com autenticação multifator (MFA) antes de permitir que eles executem ações particularmente confidenciais.

Por exemplo, você pode ter uma política que permita que um usuário execute as ações `RunInstances`, `DescribeInstances` e `StopInstances` do Amazon EC2. Mas você pode restringir uma ação destrutiva, tal como `TerminateInstances` e garantir que os usuários possam executar essa ação apenas se eles autenticarem com um dispositivo MFA da AWS.

Tópicos

- Visão geral
- Cenário: Proteção por MFA para delegação entre contas
- Cenário: Proteção por MFA para acesso às operações da API na conta atual
- Cenário: Proteção por MFA para recursos que têm políticas baseadas em recurso

Visão geral

A inclusão da proteção de MFA às operações de API envolve estas tarefas:

1. O administrador configura um dispositivo MFA da AWS para cada usuário que precisa fazer solicitações de API que exigem autenticação de MFA. Para obter mais informações, consulte AWSCódigo da autenticação multifator no IAM da.
2. O administrador cria políticas para os usuários que incluem um elemento `Condition` que verifica se o usuário autenticou com um dispositivo MFA da AWS.
3. O usuário chama uma das operações de API do AWS STS que é compatível com os parâmetros de MFA: AssumeRole ou GetSessionToken. Como parte da chamada, o usuário inclui o identificador do dispositivo que está associado a ele. O usuário também inclui a time-based one-time password (TOTP – Senha de uso único baseada em tempo) que o dispositivo gera. Em qualquer um dos casos, o usuário recebe novamente credenciais de segurança temporárias que ele pode então usar para fazer solicitações adicionais para a AWS.

> **Note**
>
> A proteção de MFA para as operações de API de um serviço está disponível apenas se o serviço for compatível com credenciais de segurança temporárias. Para obter uma lista destes serviços, consulte Uso de credenciais de segurança temporárias para acessar a AWS.

Se a autorização falhar, a AWS retornará uma mensagem de erro "Acesso negado" (como para qualquer acesso não autorizado). Com políticas de API protegidas pela MFA em vigor, a AWS nega o acesso às operações de API especificadas nas políticas se o usuário tentar chamar uma operação da API sem uma autenticação MFA válida. A operação também é negada se o time stamp da solicitação para a operação de API estiver fora do intervalo permitido especificado na política. O usuário deve ser reautenticado com MFA através da solicitação de novas credenciais de segurança temporárias com um código de MFA e número de série do dispositivo.

Políticas do IAM com condições de MFA

Políticas com condições de MFA podem ser anexadas a:

- Um usuário ou grupo do IAM
- Um recurso como um bucket do Amazon S3, uma fila do Amazon SQS ou um tópico do Amazon SNS
- A política de confiança de uma função do IAM que pode ser assumida por um usuário

Você pode usar uma condição de MFA em uma política para verificar as seguintes propriedades:

- Existência: para verificar se o usuário fez a autenticação com MFA, verifique se a chave `aws:MultiFactorAuthPresent` é `True` em uma condição `Bool`. A chave só está presente quando o usuário realiza a autenticação com credenciais de curto prazo. As credenciais de longo prazo, como as chaves de acesso, não incluem essa chave.
- Duração: se você deseja conceder acesso apenas em um período especificado após a autenticação com MFA, use um tipo de condição numérica para comparar o tempo da chave `aws:MultiFactorAuthAge` com um valor (por exemplo, 3.600 segundos). Observe que a chave `aws:MultiFactorAuthAge` não estará presente se a MFA não tiver sido usada.

O exemplo a seguir mostra a política de confiança de uma função do IAM que inclui uma condição de MFA para testar a existência de autenticação MFA. Com essa política, os usuários da Conta da AWS especificada no elemento `Principal` (substituir `ACCOUNT-B-ID` por um ID de Conta da AWS válido) podem assumir a perfil à que essa política está anexada. No entanto, esses usuários só podem assumir a função se eles forem autenticados usando a MFA.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Para obter mais informações sobre os tipos de condições para MFA, consulte AWSChaves de contexto de condições globais da, Operadores de condição numéricos e Operador de condição para verificar a existência de chaves de condição.

Escolher entre GetSessionToken e AssumeRole

O AWS STS fornece duas operações de API que permitem que os usuários passem informações de MFA: `GetSessionToken` e `AssumeRole`. A operação de API que o usuário chama para obter credenciais de segurança temporárias depende de qual dos seguintes cenários se aplica.

Use **`GetSessionToken`** nos seguintes cenários:

- Chame as operações de API que acessam recursos na mesma Conta da AWS que o usuário do IAM efetua a solicitação. Observe que credenciais temporárias de uma solicitação `GetSessionToken` poderão acessar operações de API do AWS STS e do IAM apenas se você incluir informações de MFA na solicitação de credenciais. Como credenciais temporárias retornadas por `GetSessionToken` incluem informações de MFA, você pode verificar a existência de MFA em operações de API individuais feitas pelas credenciais.
- Acesso aos recursos protegidos com políticas baseadas em recursos que incluem uma condição de MFA.

O objetivo da operação `GetSessionToken` é autenticar o usuário que usa a MFA. Não é possível usar políticas para controlar as operações de autenticação.

Use **`AssumeRole`** nos seguintes cenários:

- Chame as operações de API que acessam recursos na mesma ou em outra Conta da AWS. As chamadas de API podem incluir qualquer API do AWS STS ou do IAM. Observe que para proteger o acesso, você impõe a MFA no momento em que o usuário assume a função. As credenciais temporárias retornadas por `AssumeRole` não incluem informações de MFA no contexto, portanto

não é possível verificar a existência de operações de API individuais quanto à MFA. É por isso que você deve usar `GetSessionToken` para restringir o acesso a recursos protegidos por políticas baseadas em recursos.

> **Note**
>
> AWS CloudTrailOs logs do conterão informações de MFA quando o usuário do IAM fizer login com MFA. Se o usuário do IAM assumir um perfil do IAM, o CloudTrail também registrará em log `mfaAuthenticated: true` nos atributos `sessionContext` para ações realizadas usando o perfil assumido. Porém, o registro em log do CloudTrail é separado do que o IAM exige quando as chamadas de API são feitas com as credenciais do perfil assumido. Para saber mais, consulte Elemento userIdentity do CloudTrail.

Detalhes sobre como implementar esses cenários são fornecidos posteriormente neste documento.

Pontos importantes sobre acesso à API protegido por MFA

É importante compreender os seguintes aspectos da proteção por MFA das operações de API:

- A proteção por MFA está disponível apenas com credenciais de segurança temporárias, que devem ser obtidas com `AssumeRole` ou `GetSessionToken`.
- Você não pode usar o acesso à API protegido por MFA com credenciais de Usuário raiz da conta da AWS.
- Você não pode usar o acesso à API protegido por MFA com chaves de segurança U2F.
- Usuários federados não podem ser atribuídos a um dispositivo com MFA para uso com os serviços da AWS, portanto eles não podem acessar os recursos da AWS controlados por MFA. (Consulte o próximo ponto.)
- Outras operações de API do AWS STS que retornam credenciais temporárias não são compatíveis com MFA. Para `AssumeRoleWithWebIdentity` e `AssumeRoleWithSAML`, o usuário é autenticado por um provedor externo e a AWS não pode determinar se aquele provedor exigiu MFA. Para `GetFederationToken`, a MFA não é necessariamente associada a um usuário específico.
- Da mesma forma, credenciais de longo prazo (chaves de acesso de usuário do IAM e chaves de acesso do usuário raiz) não podem ser usadas com o acesso à API protegido por MFA, pois elas não expiram.

- `AssumeRole` e `GetSessionToken` também podem ser chamados sem informações de MFA. Neste caso, o chamador recebe as credenciais de segurança temporárias, mas as informações da sessão para essas credenciais temporárias não indicam que o usuário realizou autenticação com MFA.
- Para estabelecer a proteção por MFA para operações de API, inclua condições de MFA nas políticas. Uma política deve incluir a chave de condição `aws:MultiFactorAuthPresent` para impor o uso de MFA. Para delegação entre contas, a política de confiança da função deve incluir a chave de condição.
- Ao permitir que outra Conta da AWS acesse recursos em sua conta, a segurança dos seus recursos dependerá da configuração da conta confiável (a outra conta, não a sua). Isso ocorre mesmo quando você exige a autenticação multifator. Qualquer identidade na conta confiável que tenha permissão para criar dispositivos MFA virtuais pode construir uma solicitação de MFA para atender esta parte de sua política de confiança da função. Antes de permitir que membros de outra conta acessem seus recursos da AWS que exigem autenticação multifator, você deve garantir que o proprietário da conta confiável siga as melhores práticas de segurança. Por exemplo, a conta confiável deve restringir o acesso a operações de API confidenciais, como operações de API de gerenciamento de dispositivos MFA, a identidades confiáveis e específicas.
- Se uma política inclui uma condição de MFA, uma solicitação é negada se os usuários não tiverem realizado autenticação por MFA ou se eles fornecerem um identificador de dispositivo MFA inválido ou TOTP inválida.

Cenário: Proteção por MFA para delegação entre contas

Nesse cenário, você deseja delegar acesso a usuários do IAM em outra conta, mas apenas se os usuários forem autenticados com um dispositivo com MFA da AWS. Para obter mais informações sobre delegação entre contas, consulte Termos e conceitos das funções.

Imagine que você tenha a conta A (a conta de confiança que possui o recurso a ser acessado) com a usuária do IAM Anaya, que possui permissão de administrador. Ela quer conceder acesso ao usuário Richard na conta B (a conta confiável), mas quer ter certeza de que Richard é autenticado por MFA antes de assumir a função.

1. Na conta de confiança A, Anaya cria uma função do IAM chamada `CrossAccountRole` e define a entidade de segurança na política de confiança da função para o ID de conta da conta B. A política de confiança concede permissão para a ação `AssumeRole` do AWS STS. Anaya também adiciona uma condição de MFA à política de confiança, como no exemplo a seguir.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

2. Anaya adiciona uma política de permissões à função que especifica o que a função tem permissão para fazer. A política de permissões para uma função com proteção por MFA não é diferente de qualquer outra política de permissões para funções. O exemplo a seguir mostra a política que Anaya adiciona à função; ela permite que um usuário que a esteja assumindo execute qualquer ação do Amazon DynamoDB na tabela `Books` na conta A. Essa política também permite a ação `dynamodb:ListTables`, que é necessária para executar ações no console.

> **Note**
>
> A política de permissões não inclui uma condição de MFA. É importante compreender que a autenticação por MFA é usada apenas para determinar se um usuário pode assumir a função. Assim que o usuário assume a função, nenhuma outra verificação de MFA é realizada.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:111122223333:table/Books"
        },
        {
            "Sid": "ListTables",
```

```
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. Na conta confiável B, o administrador se certifica de que o usuário do IAM, Richard, esteja configurado com um dispositivo com MFA da AWS e de que ele saiba o ID do dispositivo. O ID do dispositivo será o número de série, se ele for um dispositivo MFA de hardware, ou o ARN do dispositivo, se ele for um dispositivo MFA virtual.
4. Na conta B, o administrador anexa a seguinte política ao usuário Richard (ou a um grupo do qual ele seja membro) que permite que ele invoque a ação `AssumeRole`. O recurso é definido como o ARN da função que Anaya criou na etapa 1. Observe que esta política não contém uma condição de MFA.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/CrossAccountRole"
            ]
        }
    ]
}
```

5. Na conta B, Richard (ou um aplicativo que Richard está executando) chama `AssumeRole`. A chamada de API inclui o ARN da função a assumir (`arn:aws:iam::`*`ACCOUNT-A-ID`*`:role/CrossAccountRole`), o ID do dispositivo MFA e a TOTP atual que Richard recebe de seu dispositivo.

   Quando Richard chama `AssumeRole`, a AWS determina se ele tem credenciais válidas, incluindo a exigência de MFA. Dessa forma, Richard assume a função com êxito e pode executar qualquer

ação do DynamoDB na tabela chamada `Books` na conta A usando as credenciais temporárias da função.

Para obter um exemplo de um programa que executa chamadas `AssumeRole`, consulte Chamar AssumeRole com autenticação MFA (Python).

Cenário: Proteção por MFA para acesso às operações da API na conta atual

Nesse cenário, você deve se certificar que um usuário na sua Conta da AWS pode acessar operações de API confidenciais apenas se ele for autenticado usando um dispositivo de MFA da AWS.

Imagine que você tem a conta A que contém um grupo de desenvolvedores que precisa trabalhar com instâncias do EC2. Desenvolvedores comuns podem trabalhar com as instâncias, mas não recebem permissões para as ações `ec2:StopInstances` ou `ec2:TerminateInstances`. Você deseja limitar essas ações privilegiadas "destrutivas" a apenas alguns usuários confiáveis, portanto você adiciona proteção por MFA à política que permite essas ações importantes do Amazon EC2.

Nesse cenário, um desses usuários confiáveis é o usuário Sofía. O usuário Anaya é um administrador na conta A.

1. Anaya se certifica de que Sofía esteja configurada com um dispositivo com MFA da AWS e de que ela saiba o ID do dispositivo. O ID do dispositivo será o número de série, se ele for um dispositivo MFA de hardware, ou o ARN do dispositivo, se ele for um dispositivo MFA virtual.
2. Anaya cria um grupo chamado `EC2-Admins` e adiciona o usuário Sofía ao grupo.
3. Anaya anexa a seguinte política ao grupo `EC2-Admins`. Essa política concede aos usuários permissão para chamar as ações `StopInstances` e `TerminateInstances` do Amazon EC2 apenas se o usuário tiver se autenticado usando a MFA.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
    ],
    "Resource": ["*"],
```

```
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.
> **Note**
>
> Para essa política entrar em vigor, os usuários devem primeiro sair e depois fazer login novamente.

   Se a usuária Sofía precisar interromper ou encerrar uma instância do Amazon EC2, ela (ou uma aplicação que ela esteja executando) chamará `GetSessionToken`. Essa operação de API transmite o ID do dispositivo MFA e a TOTP atual que Sofía recebeu de seu dispositivo.
5. A usuária Sofía (ou uma aplicação que Sofía esteja usando) usa as credenciais temporárias fornecidas por `GetSessionToken` para chamar a ação `StopInstances` ou `TerminateInstances` do Amazon EC2.

   Para obter um exemplo de um programa que executa chamadas `GetSessionToken`, consulte Chamar GetSessionToken com autenticação MFA adiante neste documento.

Cenário: Proteção por MFA para recursos que têm políticas baseadas em recurso

Nesse cenário, você é o proprietário de um bucket do S3, de uma fila do SQS ou de um tópico do SNS. Você deseja ter certeza de que qualquer usuário de qualquer Conta da AWS que acessa o recurso seja autenticado por um dispositivo de MFA da AWS.

Este cenário ilustra uma maneira de fornecer proteção por MFA entre contas sem a exigência de que os usuários assumam uma função primeiro. Neste caso, o usuário pode acessar o recurso se atender a três condições: estar autenticado por MFA, ser capaz de obter credenciais de segurança temporárias do `GetSessionToken`, e ter uma conta de confiança da política do recurso.

Imagine que você está na conta A e cria um bucket do S3. Você deseja conceder acesso a este bucket para os usuários em várias Contas da AWS diferentes, mas somente se esses usuários forem autenticados com MFA.

Nesse cenário, o usuário Anaya é um administrador na conta A. O usuário Nikhil é um usuário do IAM na conta C.

1. Na conta A, Anaya cria um bucket chamado `Account-A-bucket`.

2. Anaya adiciona a política de bucket ao bucket. A política permite que qualquer usuário na conta A, B ou C execute as ações `PutObject` e `DeleteObject` do Amazon S3 no bucket. A política inclui uma condição de MFA.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> O Amazon S3 oferece um recurso de exclusão de MFA para acesso à conta raiz (apenas). Você pode habilitar a exclusão de MFA do Amazon S3 ao definir o estado de versionamento do bucket. A exclusão de MFA do Amazon S3 não pode ser aplicada a um usuário do IAM e é gerenciada independentemente do acesso à API protegido por MFA. Um usuário do IAM com permissões para excluir um bucket não pode excluir um bucket com a exclusão de MFA do Amazon S3 habilitada. Para obter mais informações sobre a exclusão de MFA do Amazon S3, consulte Exclusão de MFA.

3. Na conta C, um administrador se certifica de que o usuário Nikhil esteja configurado com um dispositivo com MFA da AWS e que ele saiba o ID do dispositivo. O ID do dispositivo será o número de série, se ele for um dispositivo MFA de hardware, ou o ARN do dispositivo, se ele for um dispositivo MFA virtual.

4. Na conta C, Nikhil (ou um aplicativo que ele está executando) chama `GetSessionToken`. A chamada inclui o ID ou ARN do dispositivo de MFA e a TOTP atual que Nikhil recebe de seu dispositivo.
5. Nikhil (ou uma aplicação que ele esteja usando) usa as credenciais temporárias retornadas por `GetSessionToken` para chamar a ação `PutObject` do Amazon S3 para carregar um arquivo para `Account-A-bucket`.

   Para obter um exemplo de um programa que executa chamadas `GetSessionToken`, consulte Chamar GetSessionToken com autenticação MFA adiante neste documento.

   > **Note**
   >
   > As credenciais temporárias que `AssumeRole` retorna não funcionam neste caso. Embora o usuário possa fornecer informações de MFA para assumir uma função, as credenciais temporárias retornadas por `AssumeRole` não incluem as informações de MFA. Essas informações são necessárias para atender à condição de MFA na política.

Código de exemplo: Solicitação de credenciais com autenticação multifator

Os exemplos a seguir, mostram como chamar as operações `GetSessionToken` e `AssumeRole` e transmitir os parâmetros de autenticação MFA. Não é necessário ter permissão para chamar `GetSessionToken`, mas você deve ter uma política que permita a chamada `AssumeRole`. As credenciais retornadas são, então, usadas para listar todos os buckets do S3 na conta.

Chamar GetSessionToken com autenticação MFA

O exemplo a seguir mostra como chamar `GetSessionToken` e transmitir informações da autenticação MFA. As credenciais de segurança temporárias retornadas pela operação `GetSessionToken` são usadas para listar todos os buckets do S3 na conta.

A política anexada ao usuário que executa esse código (ou a um grupo em que o usuário está) fornece as permissões para as credenciais temporárias retornadas. Para este código de exemplo, a política deve conceder ao usuário permissão para solicitar a operação `ListBuckets` do Amazon S3.

Os exemplos de código a seguir mostram como usar o `GetSessionToken`.

CLI

AWS CLI

Como obter um conjunto de credenciais de curto prazo para uma identidade do IAM

O comando `get-session-token`, apresentado a seguir, recupera um conjunto de credenciais de curto prazo para a identidade do IAM que executa a chamada. As credenciais resultantes podem ser usadas para solicitações em que a autenticação multifator (MFA) é requerida pela política. As credenciais expiram 15 minutos após serem geradas.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Resultado:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Para obter mais informações, consulte Solicitação de credenciais de segurança temporárias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetSessionToken na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por um determinado período. As credenciais usadas para solicitar

credenciais temporárias são inferidas dos padrões atuais do shell. Para especificar outras credenciais, use os parâmetros -ProfileName ou -AccessKey/-SecretKey.

```
Get-STSSessionToken
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemplo 2: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora. As credenciais usadas para fazer a solicitação são obtidas do perfil especificado.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemplo 3: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora usando o número de identificação do dispositivo de MFA associado à conta cujas credenciais estão especificadas no perfil 'myprofilename' e o valor fornecido pelo dispositivo.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Consulte detalhes da API em GetSessionToken na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por um determinado período. As credenciais usadas para solicitar credenciais temporárias são inferidas dos padrões atuais do shell. Para especificar outras credenciais, use os parâmetros -ProfileName ou -AccessKey/-SecretKey.

```
Get-STSSessionToken
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemplo 2: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora. As credenciais usadas para fazer a solicitação são obtidas do perfil especificado.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
```

```
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

Exemplo 3: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora usando o número de identificação do dispositivo de MFA associado à conta cujas credenciais estão especificadas no perfil 'myprofilename' e o valor fornecido pelo dispositivo.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Saída:

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

- Para obter detalhes sobre a API, consulte GetSessionToken na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha um token de sessão passando um token de MFA e use-o para listar os buckets do Amazon S3 para a conta.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
```

```
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Consulte detalhes da API em GetSessionToken na Referência da API AWS SDK for Python (Boto3).

Chamar AssumeRole com autenticação MFA (Python)

Os exemplos a seguir mostram como chamar `AssumeRole` e transmitir informações da autenticação MFA. As credenciais de segurança temporárias retornadas por `AssumeRole` são, então, usadas para listar todos os buckets do Amazon S3 na conta.

Para ter mais informações sobre esse cenário, consulte Cenário: Proteção por MFA para delegação entre contas.

Os exemplos de código a seguir mostram como usar o `AssumeRole`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code Examples Repository.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
```

```
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
```

```
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em AssumeRole na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
```

```
                    outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como assumir um perfil

O comando `assume-role`, apresentado a seguir, recupera um conjunto de credenciais de curto prazo para o perfil do IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Resultado:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
```

```
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

A saída do comando contém uma chave de acesso, uma chave secreta e um token de sessão que você pode usar para se autenticar na AWS.

Para o uso da AWS CLI, é possível configurar um perfil nomeado associado a um perfil. Ao usar o perfil, a AWS CLI chamará assume-role e gerenciará credenciais para você. Para obter mais informações, consulte Uso de perfis do IAM na AWS CLI no Guia do usuário da AWS CLI.

- Consulte detalhes da API em AssumeRole na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
```

```
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
```

```
                    roleSessionName - An identifier for the assumed role session
(for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
exTime);
```

```
        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no Repositório de exemplos de código da AWS.

Crie o cliente.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Assuma um perfil do IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
```

```
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });
```

```
//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: retorna um conjunto de credenciais temporárias (chave de acesso, chave secreta e token de sessão) que, durante uma hora, podem ser usadas para acessar recursos da AWS aos quais o usuário solicitante normalmente não teria acesso. As credenciais retornadas têm as permissões permitidas pela política de acesso do perfil assumido e pela política fornecida

(não é possível usar a política fornecida para conceder permissões além das definidas pela política de acesso do perfil que está sendo assumido).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemplo 2: retorna um conjunto de credenciais temporárias, válidas por uma hora, que têm as mesmas permissões definidas na política de acesso do perfil que está sendo assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemplo 3: retorna um conjunto de credenciais temporárias que fornecem o número de série e o token gerado de uma MFA associada às credenciais do usuário usadas para executar o cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemplo 4: retorna um conjunto de credenciais temporárias que assumiram um perfil definido em uma conta de cliente. Para cada perfil que o terceiro possa assumir, a conta do cliente deve criar um perfil usando um identificador a ser transmitido no parâmetro -ExternalId sempre que o perfil for assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Consulte detalhes da API em AssumeRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: retorna um conjunto de credenciais temporárias (chave de acesso, chave secreta e token de sessão) que, durante uma hora, podem ser usadas para acessar recursos da AWS aos quais o usuário solicitante normalmente não teria acesso. As credenciais retornadas têm as permissões permitidas pela política de acesso do perfil assumido e pela política fornecida (não é possível usar a política fornecida para conceder permissões além das definidas pela política de acesso do perfil que está sendo assumido).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemplo 2: retorna um conjunto de credenciais temporárias, válidas por uma hora, que têm as mesmas permissões definidas na política de acesso do perfil que está sendo assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemplo 3: retorna um conjunto de credenciais temporárias que fornecem o número de série e o token gerado de uma MFA associada às credenciais do usuário usadas para executar o cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemplo 4: retorna um conjunto de credenciais temporárias que assumiram um perfil definido em uma conta de cliente. Para cada perfil que o terceiro possa assumir, a conta do cliente deve criar um perfil usando um identificador a ser transmitido no parâmetro -ExternalId sempre que o perfil for assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Para obter detalhes sobre a API, consulte AssumeRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Assuma um perfil do IAM que exija um token de MFA e use credenciais temporárias para listar os buckets do Amazon S3 para a conta.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
```

```
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
```

```
        Err(e) => println!("{:?}", e),
    }
}
```

- Consulte detalhes da API em AssumeRole na Referência do AWS SDK para API Rust.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte AssumeRole na Referência do AWS SDK para API Swift.

Credenciais específicas do serviço para usuários do IAM

Credenciais específicas do serviço são mecanismos de autenticação especializados criados para serviços específicos da AWS. Essas credenciais fornecem autenticação simplificada em comparação com as credenciais padrão da AWS e são ajustadas aos requisitos de autenticação de serviços individuais da AWS. Diferentemente das chaves de acesso, que podem ser usadas em vários serviços da AWS, as credenciais específicas do serviço destinam-se a ser usadas apenas com o serviço para o qual foram criadas. Essa abordagem direcionada aumenta a segurança por limitar o escopo das credenciais.

As credenciais específicas do serviço geralmente consistem em um par de nome de usuário e senha ou em chaves de API especializadas que são formatadas de acordo com os requisitos do serviço específico. Quando você cria credenciais específicas do serviço, elas estão ativas por padrão e podem ser usadas imediatamente. Você pode ter, no máximo, dois conjuntos de credenciais específicas do serviço para cada serviço compatível por usuário do IAM. Esse limite permite que você mantenha um conjunto ativo enquanto troca para um novo conjunto quando necessário. Atualmente, a AWS é compatível com credenciais específicas dos seguintes serviços:

Quando usar credenciais específicas do serviço

As credenciais específicas do serviço são destinadas à compatibilidade com bibliotecas, SDKs, ferramentas ou aplicações de terceiros que não são compatíveis nativamente com credenciais da AWS, AWS SDKs ou APIs da AWS. Esses casos de uso incluem migrar para a AWS serviços existentes em infraestrutura auto-hospedada ou serviços hospedados por outros provedores.

Ao começar do zero, e sempre que possível, recomendamos usar credenciais da AWS temporárias, como as fornecidas por um perfil do IAM, para se autenticar em um serviço da AWS usando um AWS SDK ou uma biblioteca que ofereça suporte a credenciais temporárias da AWS.

Troca de credenciais específicas do serviço

Como uma prática recomendada de segurança, troque as credenciais específicas do serviço regularmente. Para trocar as credenciais sem prejudicar as aplicações:

1. Crie um segundo conjunto de credenciais específicas do serviço para o mesmo serviço e usuário do IAM

2. Atualize todas a aplicações para que usem as novas credenciais e verifique se elas funcionam bem
3. Altere o status das credenciais originais para "Inativas"
4. Verifique se todas as aplicações continuam a funcionar adequadamente
5. Exclua as credenciais específicas do serviço inativas quando tiver certeza de que não são mais necessárias

Monitorar credenciais específicas do serviço

Você pode usar o AWS CloudTrail para monitorar o uso de credenciais específicas do serviço em sua conta da AWS. Para visualizar os eventos do CloudTrail relacionados ao uso das credenciais específicas do serviço, revise os logs do CloudTrail para verificar os eventos do serviço nos quais as credenciais são usadas. Para obter mais informações, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.

Para ter mais segurança, pense em configurar alarmes do CloudWatch que notifiquem sobre padrões específicos de uso das credenciais que possam indicar acesso não autorizado ou outras preocupações de segurança. Para obter mais informações, consulte Monitoring CloudTrail Log Files with Amazon CloudWatch Logs no AWS CloudTrail User Guide.

Os tópicos a seguir fornecem informações sobre credenciais específicas do serviço.

Tópicos

- Chaves de API para os serviços da AWS
- Usar o IAM com o Amazon Keyspaces (para Apache Cassandra)

Chaves de API para os serviços da AWS

Alguns serviços da AWS oferecem suporte a chaves de API para autenticar solicitações programáticas, além das credenciais padrão do IAM, como credenciais de segurança temporárias e chaves de acesso de longo prazo. A AWS oferece dois tipos de chaves de API:

- Chaves de API de longo prazo: as chaves de API de longo prazo são associadas a um usuário do IAM e são geradas usando as credenciais específicas do serviço do IAM. Essas credenciais são projetadas para uso com apenas um único serviço da AWS, aumentando a segurança ao limitar o escopo das credenciais. Você pode definir um prazo de validade para a API de longo prazo. Para

gerar chaves de API de longo prazo, você pode usar o console do IAM ou o console específico do serviço, a AWS CLI ou AWS API.

- Chaves de API de curto prazo: uma chave de API de curto prazo é uma URL pré-assinada que usa o AWS Signature versão 4. As chaves de API de curto prazo têm as mesmas permissões e validade que as credenciais da identidade que gera a chave de API e são válidas por até 12 horas ou pelo tempo restante da sua sessão do console, o que for menor. Você pode usar o console do Amazon Bedrock/Claude Platform na AWS, Python e pacotes de outras linguagens de programação para gerar chaves de API de curto prazo. Para obter mais informações, consulte Gerar chaves de API do Amazon Bedrock para facilitar o acesso à API do Amazon Bedrock no Guia do usuário do Amazon Bedrock e Autenticação no Guia do usuário do Claude Platform na AWS.

> **Note**
>
> As chaves de API de longo prazo têm um risco de segurança maior em comparação com as chaves de API de curto prazo. Recomendamos o uso de chaves de API de curto prazo ou de credenciais de segurança temporárias quando possível. Se você usar chaves de API de longo prazo, recomendamos que implemente práticas regulares de troca das chaves.

Serviços que oferecem suporte a chaves de API

A tabela a seguir lista os serviços da AWS que oferecem suporte às chaves de API e o tipo de chave de API compatível com cada serviço.

#	Serviço	Chaves de API de longo prazo	Chaves de API de curto prazo	Política gerenciada anexada automaticamente	Documentação específica do serviço
1	Amazon Bedrock	Sim	Sim	AmazonBedrockLimitedAccess	Usar uma chave de API do Amazon Bedrock

#	Serviço	Chaves de API de longo prazo	Chaves de API de curto prazo	Política gerenciada anexada automaticamente	Documentação específica do serviço
2	Claude Platform na AWS	Sim	Sim	AnthropicInferenceAccess	autenticação do
3	Amazon CloudWatch	Sim	N/D	CloudWatchAPIKeyAccess	Configurar autenticação do token do portador para Metrics
4	Amazon CloudWatch Logs	Sim	N/D	CloudWatchLogsAPIKeyAccess	Configurar autenticação do token do portador

Quando você gera uma chave de API de longo prazo para um serviço, a política gerenciada da AWS correspondente é automaticamente anexada ao usuário do IAM, concedendo acesso às principais operações do serviço. Se você precisar de acesso adicional, poderá modificar as permissões do usuário do IAM. Para obter mais informações sobre modificação de permissões, consulte the section called “Adicionar ou remover permissões de identidade”.

Para saber mais sobre as chaves de API para serviços específicos, consulte os links da documentação específica do serviço na tabela acima.

Pré-requisitos para chaves de API de longo prazo

Antes de gerar uma chave de API de longo prazo no console do IAM, você deve atender aos seguintes pré-requisitos:

- Um usuário do IAM a ser associado à chave de API de longo prazo. Para obter instruções para criar um usuário do IAM, consulte the section called “Criar um usuário”.

- Você deve ter as seguintes permissões da política do IAM para gerenciar as credenciais específicas do serviço de um usuário do IAM. O exemplo de política concede permissão para criar, listar, atualizar, excluir e redefinir credenciais específicas do serviço. Substitua o valor *username* no elemento Recurso pelo nome do usuário do IAM para o qual você vai gerar as chaves de API de longo prazo:

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageBedrockServiceSpecificCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:UpdateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ResetServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/username"
        }
    ]
}
```

Como gerar uma chave de API de longo prazo (console)

Para gerar uma chave de API de longo prazo para um serviço específico no console do IAM

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Usuários.
3. Selecione o usuário do IAM para o qual você deseja gerar as chaves de API de longo prazo.
4. Selecione a guia Credenciais de segurança.
5. Na seção Chaves de API , escolha Gerar chave de API.
6. Na lista suspensa de serviços da AWS, escolha o serviço no qual você deseja que a chave de API seja autenticada.
7. Em Validade da chave de API, faça uma das seguintes opções:

- Escolha a duração da validade da chave de API de 1, 5, 30, 90 ou 365 dias.
- Escolha Duração personalizada para especificar uma data de validade personalizada para a chave de API.
- Escolha Nunca expira (não recomendado).

8. Escolha Gerar chave de API.
9. Copie ou baixe a chave de API. Esta é a única vez que você pode visualizar o valor da chave de API.

> **Important**
>
> Armazene a chave de API em segurança. Após fechar a caixa de diálogo, não será mais possível recuperar a chave de API. Se você perder ou esquecer sua chave de API, não poderá recuperá-la. Em vez disso, gere uma nova chave de API e desative a chave antiga.

Como gerar uma chave de API de longo prazo (AWS CLI)

Para gerar uma chave de API de longo prazo usando a AWS CLI, siga as seguintes etapas:

1. Crie um usuário do IAM que será usado com o servido via comando create-user:

```
aws iam create-user \
    --user-name APIKeyUser_1
```

2. Anexe a política gerenciada pela AWS ao usuário do IAM usando o comando attach-user-policy.

   Para Amazon Bedrock:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AmazonBedrockLimitedAccess
```

   Para Claude Platform na AWS:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/AnthropicInferenceAccess
```

   Para Amazon CloudWatch:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchAPIKeyAccess
```

Para Amazon CloudWatch Logs:

```
aws iam attach-user-policy --user-name APIKeyUser_1 \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchLogsAPIKeyAccess
```

3. Gere a chave de API de longo prazo usando o comando create-service-specific-credential.

Para Amazon Bedrock:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name bedrock.amazonaws.com \
    --credential-age-days 30
```

Para Claude Platform na AWS:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name aws-external-anthropic.amazonaws.com \
    --credential-age-days 30
```

Para Amazon CloudWatch:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name cloudwatch.amazonaws.com \
    --credential-age-days 30
```

Para Amazon CloudWatch Logs:

```
aws iam create-service-specific-credential \
    --user-name APIKeyUser_1 \
    --service-name logs.amazonaws.com \
    --credential-age-days 30
```

> **Note**
>
> O parâmetro `--credential-age-days` é opcional. Você pode especificar um valor entre 1 e 36.600 dias. Se você omitir esse parâmetro, a chave de API não expirará.

O valor `ServiceApiKeyValue` exibido na resposta é a chave de API de longo prazo do respectivo serviço. Armazene o valor `ServiceApiKeyValue` em segurança, pois não será possível recuperá-lo mais tarde.

Listar chaves de API de longo prazo (AWS CLI)

Para listar os metadados das chaves de API de longo prazo para um usuário específico, use o comando list-service-specific-credentials com o parâmetro `--user-name`:

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --user-name APIKeyUser_1
```

> **Note**
>
> Substitua `bedrock.amazonaws.com` pelo nome do serviço apropriado (por exemplo, `logs.amazonaws.com` para Amazon CloudWatch Logs ou `aws-external-anthropic.amazonaws.com` para Claude Platform na AWS).

Para listar todos os metadados das chaves de API de longo prazo na conta, use o comando list-service-specific-credentials com o parâmetro `--all-users`:

```
aws iam list-service-specific-credentials \
    --service-name bedrock.amazonaws.com \
    --all-users
```

Atualizar o status de uma chave de API de longo prazo (AWS CLI)

Para atualizar o status de uma chave de API de longo prazo, use o comando update-service-specific-credential:

```
aws iam update-service-specific-credential \
```

```
    --user-name "APIKeyUser_1" \
    --service-specific-credential-id "ACCA1234EXAMPLE1234" \
    --status Inactive|Active
```

Como gerar uma chave de API de longo prazo (API da AWS)

Você pode usar as seguintes operações de API do IAM para gerenciar as chaves de API de longo prazo de qualquer serviço compatível:

- CreateServiceSpecificCredential
- ListServiceSpecificCredentials
- UpdateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential

Chaves de API de curto prazo (serviços específicos)

No momento, as chaves de API de curto prazo são aceitas somente por serviços específicos.

Para obter informações sobre como gerar e usar chaves de API de curto prazo com o Amazon Bedrock, consulte Gerar uma chave de API no Guia do usuário do Amazon Bedrock.

Para obter informações sobre como gerar e usar chaves de API de curto prazo para Claude Platform na AWS, consulte Autenticação no Guia do usuário do Claude Platform na AWS.

Informações específicas do serviço

- Para obter mais informações sobre como usar uma chave do Amazon Bedrock, consulte Usar uma chave de API do Amazon Bedrock no Guia do usuário do Amazon Bedrock.
- Para obter mais informações sobre o uso de chaves de API com o Claude Platform na AWS, consulte Autenticação no Guia do usuário do Claude Platform na AWS.
- Para obter mais informações sobre o uso de chaves de API com o Amazon CloudWatch, consulte Configurar a autenticação do token do portador para Metrics no Guia do usuário do Amazon CloudWatch.
- Para obter mais informações sobre o uso de chaves de API com o Amazon CloudWatch Logs, consulte Configurar a autenticação do token do portador no Guia do usuário do Amazon CloudWatch Logs.

Usar o IAM com o Amazon Keyspaces (para Apache Cassandra)

O Amazon Keyspaces (para Apache Cassandra) é um serviço de banco de dados compatível com Apache Cassandra, escalável, de alta disponibilidade e gerenciado. Você pode acessar o Amazon Keyspaces por meio do Console de gerenciamento da AWS ou de maneira programática. Para acessar o Amazon Keyspaces de maneira programática com credenciais específicas do serviço, você pode usar `cqlsh` ou drivers de código aberto do Cassandra. Credenciais específicas do serviço incluem um nome de usuário e senha como os que o Cassandra usa para autenticação e gerenciamento de acesso. Você pode ter no máximo dois conjuntos de credenciais específicas do serviço para cada serviço suportado por usuário.

Para acessar o Amazon Keyspaces de maneira programática com chaves de acesso da AWS, você pode usar o AWS SDK, a AWS Command Line Interface (AWS CLI) ou drivers do Cassandra de código aberto com o plug-in SigV4. Para saber mais, consulte Create and configure AWS credentials for Amazon Keyspaces no Amazon Keyspaces (for Apache Cassandra) Developer Guide.

> **Note**
>
> Se você planeja interagir com o Amazon Keyspaces apenas por meio do console, não precisa gerar credenciais específicas do serviço. Para obter mais informações, consulte Accessing Amazon Keyspaces using the console (Acessar o Amazon Keyspaces usando o console) no Amazon Keyspaces (para Apache Cassandra) Developer Guide (Guia do desenvolvedor do Amazon Keyspaces [for Apache Cassandra]).

Para obter mais informações sobre as permissões necessárias para acessar o Amazon Keyspaces, consulte Exemplos de políticas baseadas em identidade do Amazon Keyspaces (para Apache Cassandra) no Guia do desenvolvedor do Amazon Keyspaces (para Apache Cassandra).

Gerar credenciais do Amazon Keyspaces (console)

É possível usar o Console de gerenciamento da AWS para gerar credenciais do Amazon Keyspaces (para Apache Cassandra) para os usuários do IAM.

Como gerar credenciais específicas do serviço Amazon Keyspaces (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação, selecione Users (Usuários) e escolha o nome do usuário que requer as credenciais.
3. Na guia Security Credentials (Credenciais de segurança), abaixo de Credentials for Amazon Keyspaces (for Apache Cassandra) (Credenciais do Amazon Keyspaces [for Apache Cassandra]), selecione Generate credentials (Gerar credenciais).
4. As credenciais específicas do seu serviço agora estão disponíveis. Esta é a única vez que a senha pode ser visualizada ou baixada. Não será possível recuperá-la posteriormente. No entanto, é possível redefinir a senha a qualquer momento. Salve o usuário e a senha em um local seguro, pois você precisará deles mais tarde.

Gerar credenciais do Amazon Keyspaces (AWS CLI)

É possível usar o AWS CLI para gerar credenciais do Amazon Keyspaces (para Apache Cassandra) para os usuários do IAM.

Para gerar credenciais específicas do serviço Amazon Keyspaces (AWS CLI)

- Use o seguinte comando:
  - aws iam create-service-specific-credential

Gerar credenciais do Amazon Keyspaces (API da AWS)

Você pode usar a API da AWS para gerar credenciais do Amazon Keyspaces (para Apache Cassandra) para seus usuários do IAM.

Para gerar credenciais específicas do serviço Amazon Keyspaces (API da AWS)

- Conclua a seguinte operação:
  - CreateServiceSpecificCredential

Encontrar credenciais da AWS não utilizadas

Para aumentar a segurança da sua Conta da AWS remova as credenciais de usuários do IAM (ou seja, senhas e chaves de acesso) que não são necessárias. Por exemplo, quando os usuários deixam sua organização ou não precisam mais de acesso à AWS, localize as credenciais que eles estavam utilizando e certifique-se de que elas não estejam mais operando. O ideal é que você exclua

as credenciais se não forem mais necessárias. Você pode sempre recriá-las em posteriormente, se necessário. No mínimo, você deve alterar a senha ou desativar as chaves de acesso para que os usuários antigos não possam mais ter acesso.

Naturalmente, a definição de não usada pode variar e geralmente significa uma credencial que não foi usada durante determinado período.

Encontrar senhas não utilizadas

Você pode usar o Console de gerenciamento da AWS para visualizar informações de uso de senha para seus usuários. Se você tiver um grande número de usuários, você pode usar o console para fazer download de um relatório de credenciais com informações sobre quando cada usuário usou sua senha do console pela última vez. Você também pode acessar as informações da AWS CLI ou da API do IAM.

Para encontrar as senhas não utilizadas (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Se necessário, adicione a coluna Último login no console na tabela dos usuários:
   a. Acima da tabela, no canto direito, selecione o ícone de configurações ( ).
   b. Em Selecionar colunas visíveis, selecione Último login do console.
   c. Escolha Confirmar para retornar à lista de usuários.
4. A coluna Console last sign-in (Último login no console) exibe o número de dias desde que o usuário efetuou login na AWS pelo console pela última vez. Você pode usar essas informações para localizar os usuários com senhas que não fizeram login há mais tempo do que o período especificado. A coluna exibe Nunca para usuários com senhas que nunca efetuaram login. Nenhuma indica usuários sem senhas. Senhas que não tenham sido usadas recentemente podem ser bons candidatos para remoção.

> **Important**
>
> Devido a um problema de serviço, os dados usados mais recentemente da senha não incluem o uso de senha entre 3 de maio de 2018 22:50 PDT e 23 de maio de 2018 14:08 PDT. Isso afeta as datas de último login mostradas no console do IAM e as datas de

> última senha usadas no relatório de credencial do IAM e retornadas pela operação da API GetUser. Se os usuários fizerem login durante a hora afetada, a data usada pela última vez na senha retornada será a data em que o usuário fez login pela última vez antes de 3 de maio de 2018. Para usuários que fizeram login depois de 23 de maio de 2018 14:08 PDT, a data usada mais recentemente para a senha retornada será precisa. Se você usar as informações usadas mais recentemente para a senha para identificar credenciais não utilizadas para exclusão, como excluir usuários que não fizeram login na AWS nos últimos 90 dias, será recomendável ajustar a janela de avaliação para incluir datas após 23 de maio de 2018. Como alternativa, se os usuários usarem chaves de acesso para acessar a AWS de maneira programática, você poderá consultar as informações usadas mais recentemente da chave de acesso, pois elas são precisas para todas as datas.

Para encontrar senhas não utilizadas ao fazer download do relatório de credenciais (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Relatório de credenciais.
3. Selecione Fazer download do relatório para fazer download de um arquivo de valores separados por vírgula (CSV) chamado `status_reports_<date>T<time>.csv`. A quinta coluna é a coluna `password_last_used` com as datas ou uma das seguintes opções:
   - N/A (N/D): os usuários que não têm qualquer senha atribuída.
   - no_information: os usuários que não usaram suas senhas desde que o IAM começou a rastrear o tempo de duração das senhas em 20 de outubro de 2014.

Para encontrar senhas não usadas (AWS CLI)

Execute o seguinte comando para encontrar senhas não usadas:

- `aws iam list-users` retorna uma lista de usuários, cada um com um valor `PasswordLastUsed`. Se o valor estiver ausente, o usuário não tem senha ou a senha não foi usada desde que o IAM começou a rastrear o tempo de duração das senhas em 20 de outubro de 2014.

Para encontrar senhas não usadas (API da AWS)

Chame a seguinte operação para encontrar senhas não usadas:

- `ListUsers` retorna uma coleção de usuários, cada um com um valor `<PasswordLastUsed>`. Se o valor estiver ausente, o usuário não tem senha ou a senha não foi usada desde que o IAM começou a rastrear o tempo de duração das senhas em 20 de outubro de 2014.

Para obter informações sobre os comandos para fazer download do relatório de credenciais, consulte Obter relatórios de credenciais (AWS CLI).

Encontrar chaves de acesso não utilizadas

Você pode usar o Console de gerenciamento da AWS para visualizar informações de uso de chaves de acesso para seus usuários. Se você tiver um grande número de usuários, você pode usar o console para fazer download de um relatório de credenciais para descobrir quando cada usuário usou sua chave de acesso pela última vez. Você também pode acessar as informações da AWS CLI ou da API do IAM.

Para localizar chaves de acesso não utilizadas (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Se necessário, adicione a coluna Chave de acesso usada pela última vez na tabela dos usuários:
   a. Acima da tabela, no canto direito, selecione o ícone de configurações ( ).
   b. Em Selecionar colunas visíveis, selecione Última chave de acesso utilizada.
   c. Escolha Confirmar para retornar à lista de usuários.
4. A coluna Chave de acesso usada pela última vez exibe o número de dias desde que o usuário acessou a AWS de forma programática pela última vez. Você pode usar essas informações para localizar os usuários com chaves de acesso que não tenham sido usadas há mais tempo do que o período especificado. A coluna exibe – para usuários sem chaves de acesso. Chaves de acesso que não tenham sido usadas recentemente podem ser bons candidatos para remoção.

Para encontrar chaves de acesso não utilizadas ao fazer download do relatório de credenciais (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Relatório de credenciais.
3. Selecione Fazer download do relatório para fazer download de um arquivo de valores separados por vírgula (CSV) chamado `status_reports_<date>T<time>.csv`. As colunas 11 a 13 contêm a última data usada, a região e informações de serviço para a chave de acesso 1. As colunas 16 a 18 contêm as mesmas informações para chave de acesso 2. O valor será N/A (N/D) se o usuário não tiver uma chave de acesso ou se o usuário não tiver usado a chave de acesso desde que o IAM começou a rastrear o tempo de duração das chaves de acesso em 22 de abril de 2015.

Para localizar chaves de acesso não utilizadas (AWS CLI)

Execute os seguintes comandos para encontrar chaves de acesso não utilizadas:

- `aws iam list-access-keys` retorna informações sobre as chaves de acesso para um usuário, incluindo o `AccessKeyID`.
- `aws iam get-access-key-last-used` exige um ID de chave de acesso e retorna a saída que inclui a `LastUsedDate`, `Region` na qual a chave de acesso foi usada pela última vez e o `ServiceName` do último serviço solicitado. Se `LastUsedDate` estiver ausente, significa que a chave de acesso não foi usada desde que o IAM começou a rastrear o tempo de duração das chaves de acesso em 22 de abril de 2015.

Para localizar chaves de acesso não utilizadas (API da AWS)

Chame as seguintes operações para encontrar chaves de acesso não utilizadas:

- `ListAccessKeys` retorna uma lista de valores `AccessKeyID` para chaves de acesso associadas ao usuário especificado.
- `GetAccessKeyLastUsed` exige um ID de chave de acesso e retorna um conjunto de valores. Os valores incluem a `LastUsedDate`, a `Region` em que a chave de acesso foi usada pela última vez e o `ServiceName` do último serviço solicitado. Se o valor estiver ausente, significa que o usuário não possui uma chave de acesso ou a chave de acesso não é usada desde que o IAM começou a rastrear o tempo de duração das chaves de acesso em 22 de abril de 2015.

Para obter informações sobre os comandos para fazer download do relatório de credenciais, consulte Obter relatórios de credenciais (AWS CLI).

Gerar relatórios de credenciais para sua Conta da AWS

Você pode gerar e fazer o download de um relatório de credenciais que lista todos os usuários em sua conta e o status de diversas credenciais deles, incluindo senhas, chaves de acesso e dispositivos MFA. Você pode obter um relatório de credenciais do Console de gerenciamento da AWS, dos SDKs da AWS e das Ferramentas da linha de comando ou da API do IAM.

> **Note**
>
> O relatório de credenciais do IAM inclui apenas as seguintes credenciais gerenciadas pelo IAM: senhas, as duas primeiras chaves de acesso por usuário, dispositivos de MFA e certificados de assinatura X.509. O relatório não inclui as credenciais específicas do serviço (como senhas do CodeCommit, chaves de API do Amazon Bedrock ou chaves de API de longo prazo do Amazon CloudWatch Logs) ou qualquer outra chave de acesso de usuário além das duas primeiras. Para obter visibilidade completa das credenciais, use as APIs ListServiceSpecificCredentials e ListAccessKeys.

Você pode usar os relatórios de credenciais para auxiliar em seus esforços de auditoria e compatibilidade. É possível usar o relatório para auditar os efeitos dos requisitos de ciclo de vida da credencial, como a atualização da chave de acesso e da senha. Você pode fornecer o relatório a um auditor externo ou conceder permissões a um auditor para que ele possa fazer download do relatório diretamente.

Você pode gerar um relatório de credenciais a cada quatro horas. Quando você solicita um relatório, o IAM primeiro verifica se um relatório da Conta da AWS foi gerado nas últimas quatro horas. Se esse for o caso, o relatório mais recente será baixado. Se o relatório mais recente da conta tiver mais de quatro horas ou se não houver relatórios anteriores para a conta, o IAM gerará e baixará um novo relatório.

Tópicos

- Permissões obrigatórias
- Noções básicas sobre o formato do relatório
- Obter relatórios de credenciais (console)

- Obter relatórios de credenciais (AWS CLI)
- Obter relatórios de credenciais (API da AWS)

Permissões obrigatórias

As seguintes permissões são necessárias para criar e fazer download de relatórios:

- Para criar um relatório de credenciais: `iam:GenerateCredentialReport`
- Para fazer download do relatório: `iam:GetCredentialReport`

Noções básicas sobre o formato do relatório

Os relatórios de credenciais são formatados como arquivos de valores separados por vírgulas (CSV). Você pode abrir arquivos CSV com software de planilha comum para realizar a análise, ou pode criar um aplicativo que consuma os arquivos CSV de forma programática e realize análises personalizadas.

O arquivo CSV contém as seguintes colunas:

usuário

O nome amigável do usuário.

arn

O nome de recurso da Amazon (ARN) do usuário. (Para obter mais informações sobre ARNs, consulte) ARNs do IAM.

user_creation_time

A data e a hora em que o usuário foi criado, no formato de data/hora ISO 8601.

password_enabled

Quando o usuário tem uma senha, esse valor será `TRUE`. Caso contrário, o valor será `FALSE`. Esse valor é `FALSE` para novas contas de membros criadas como parte da organização, pois elas não têm, por padrão, credenciais de usuário-raiz.

password_last_used

A data e a hora em que a senha do usuário ou do Usuário raiz da conta da AWS foi usada pela última vez para fazer login em um site da AWS, no formato de data/hora ISO 8601. Os sites da

AWS que registram a hora do último login de um usuário são o Console de gerenciamento da AWS, os fóruns de discussão da AWS e o AWS Marketplace. Quando uma senha é usada mais de uma vez em um intervalo de 5 minutos, apenas o primeiro uso é gravado nesse campo.

- O valor nesse campo é `no_information` nos seguintes casos:
  - A senha do usuário nunca foi usada.
  - Não há dados de login associados à senha, como quando a senha do usuário não tiver sido usada depois que o IAM começou a rastrear essas informações em 20 de outubro de 2014.
- O valor nesse campo será `N/A` (não aplicável) quando o usuário não tiver uma senha.

> **Important**
>
> Devido a um problema de serviço, os dados usados mais recentemente da senha não incluem o uso de senha entre 3 de maio de 2018 22:50 PDT e 23 de maio de 2018 14:08 PDT. Isso afeta as datas de último login mostradas no console do IAM e as datas de última senha usadas no relatório de credencial do IAM e retornadas pela operação da API GetUser. Se os usuários fizerem login durante a hora afetada, a data usada pela última vez na senha retornada será a data em que o usuário fez login pela última vez antes de 3 de maio de 2018. Para usuários que fizeram login depois de 23 de maio de 2018 14:08 PDT, a data usada mais recentemente para a senha retornada será precisa.
> Se você usar as informações usadas mais recentemente para a senha para identificar credenciais não utilizadas para exclusão, como excluir usuários que não fizeram login na AWS nos últimos 90 dias, será recomendável ajustar a janela de avaliação para incluir datas após 23 de maio de 2018. Como alternativa, se os usuários usarem chaves de acesso para acessar a AWS de maneira programática, você poderá consultar as informações usadas mais recentemente da chave de acesso, pois elas são precisas para todas as datas.

password_last_changed

A data e a hora em que a senha do usuário foi definida pela última vez no formato de data/hora ISO 8601. Se o usuário não tiver uma senha, o valor nesse campo será `N/A` (não aplicável).

password_next_rotation

Quando a conta tem uma política de senha que requer a mudança de senha, esse campo contém a data e a hora em formato de data/hora ISO 8601, quando o usuário precisa definir uma nova senha. O valor para a Conta da AWS (raiz) é sempre `not_supported`.

mfa_active

Quando um dispositivo de autenticação multifator (MFA) for ativado para o usuário, esse valor será `TRUE`. Caso contrário, o valor será `FALSE`.

access_key_1_active

Quando o usuário tem uma chave de acesso e o status da chave de acesso é `Active`, esse valor será `TRUE`. Caso contrário, o valor será `FALSE`. Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

access_key_1_last_rotated

A data e a hora, em formato de data/hora ISO 8601, quando a chave de acesso do usuário foi criada ou alterada pela última vez. Se o usuário não tiver uma chave de acesso ativa, o valor nesse campo será `N/A` (não aplicável). Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

access_key_1_last_used_date

A data e a hora, em formato de data/hora ISO 8601, quando a chave de acesso do usuário tiver sido usada recentemente para assinar uma solicitação da API da AWS. Quando uma chave de acesso tiver sido usada mais de uma vez em um intervalo de 15 minutos, apenas o primeiro uso será gravado nesse campo. Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

O valor nesse campo será `N/A` (não aplicável) nos seguintes casos:

- O usuário não tiver uma chave de acesso.
- A chave de acesso nunca tiver sido usada.
- A chave de acesso não tiver sido usada depois que o IAM começou a rastrear essas informações em 22 de abril de 2015.

access_key_1_last_used_region

A região da AWS em que a chave de acesso foi usada mais recentemente. Quando uma chave de acesso tiver sido usada mais de uma vez em um intervalo de 15 minutos, apenas o primeiro uso será gravado nesse campo. Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

O valor nesse campo será `N/A` (não aplicável) nos seguintes casos:

- O usuário não tiver uma chave de acesso.
- A chave de acesso nunca tiver sido usada.
- A chave de acesso tiver sido usada pela última vez antes do IAM começar a rastrear essas informações em 22 de abril de 2015.

- O último serviço usado não for específico da região, como o Amazon S3.

access_key_1_last_used_service

O serviço da AWS que foi acessado recentemente com a chave de acesso. O valor nesse campo usa o namespace do serviço, por exemplo, `s3` para o Amazon S3 e `ec2` para o Amazon EC2. Quando uma chave de acesso tiver sido usada mais de uma vez em um intervalo de 15 minutos, apenas o primeiro uso será gravado nesse campo. Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

O valor nesse campo será `N/A` (não aplicável) nos seguintes casos:

- O usuário não tiver uma chave de acesso.
- A chave de acesso nunca tiver sido usada.
- A chave de acesso tiver sido usada pela última vez antes do IAM começar a rastrear essas informações em 22 de abril de 2015.

access_key_2_active

Quando o usuário tem uma segunda chave de acesso e o status da segunda chave de acesso é `Active`, esse valor será `TRUE`. Caso contrário, o valor será `FALSE`. Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

> **Note**
>
> Os usuários podem ter até duas chaves de acesso para facilitar a rotação atualizando a chave primeiro e depois excluindo a chave anterior. Para obter mais informações sobre a atualização de chaves de acesso, consulte Atualizar chaves de acesso.

access_key_2_last_rotated

A data e a hora, no formato de data/hora ISO 8601, quando a segunda chave de acesso do usuário tiver sido criada ou alterada mais recentemente. Se o usuário não tiver uma segunda chave de acesso ativa, o valor nesse campo será `N/A` (não aplicável). Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

access_key_2_last_used_date

A data e a hora, em formato de data/hora ISO 8601, quando a segunda chave de acesso do usuário tiver sido usada recentemente para assinar uma solicitação da API da AWS. Quando uma chave de acesso tiver sido usada mais de uma vez em um intervalo de 15 minutos, apenas o

primeiro uso será gravado nesse campo. Aplica-se tanto a usuários do IAM como usuários-raiz da conta.

O valor nesse campo será `N/A` (não aplicável) nos seguintes casos:

- O usuário não tiver uma segunda chave de acesso.
- A segunda chave de acesso do usuário nunca tiver sido usada.
- A segunda chave de acesso do usuário tiver sido usada pela última vez antes de o IAM começar a rastrear essas informações em 22 de abril de 2015.

access_key_2_last_used_region

A região da AWS em que a segunda chave de acesso do usuário foi usada mais recentemente. Quando uma chave de acesso tiver sido usada mais de uma vez em um intervalo de 15 minutos, apenas o primeiro uso será gravado nesse campo. Aplica-se tanto a usuários do IAM como usuários-raiz da conta. O valor nesse campo será `N/A` (não aplicável) nos seguintes casos:

- O usuário não tiver uma segunda chave de acesso.
- A segunda chave de acesso do usuário nunca tiver sido usada.
- A segunda chave de acesso do usuário tiver sido usada pela última vez antes de o IAM começar a rastrear essas informações em 22 de abril de 2015.
- O último serviço usado não for específico da região, como o Amazon S3.

access_key_2_last_used_service

O serviço da AWS que foi acessado recentemente com a segunda chave de acesso do usuário. O valor nesse campo usa o namespace do serviço, por exemplo, `s3` para o Amazon S3 e `ec2` para o Amazon EC2. Quando uma chave de acesso tiver sido usada mais de uma vez em um intervalo de 15 minutos, apenas o primeiro uso será gravado nesse campo. Aplica-se tanto a usuários do IAM como usuários-raiz da conta. O valor nesse campo será `N/A` (não aplicável) nos seguintes casos:

- O usuário não tiver uma segunda chave de acesso.
- A segunda chave de acesso do usuário nunca tiver sido usada.
- A segunda chave de acesso do usuário tiver sido usada pela última vez antes de o IAM começar a rastrear essas informações em 22 de abril de 2015.

cert_1_active

Quando o usuário tem um certificado de assinatura X.509 e o status do certificado é `Active`, esse valor será `TRUE`. Caso contrário, o valor será `FALSE`.

cert_1_last_rotated

A data e a hora, em formato de data/hora ISO 8601, quando o certificado de assinatura do usuário foi criado ou alterado pela última vez. Se o usuário não tiver um certificado de assinatura ativo, o valor nesse campo será `N/A` (não aplicável).

cert_2_active

Quando o usuário tem um segundo certificado de assinatura X.509 e o status do certificado é `Active`, esse valor será `TRUE`. Caso contrário, o valor será `FALSE`.

> **Note**
>
> Os usuários podem ter até dois certificado de assinatura X.509 para facilitar a mudança do certificado.

cert_2_last_rotated

A data e a hora, em formato de data/hora ISO 8601, quando o segundo certificado de assinatura do usuário foi criado ou alterado pela última vez. Se o usuário não tiver um segundo certificado de assinatura ativo, o valor nesse campo será `N/A` (não aplicável).

additional_credentials_info

Quando o usuário tem mais de duas chaves de acesso ou certificados, esse valor é o número de chaves de acesso ou certificados adicionais e as ações que você pode usar para listar as chaves de acesso ou os certificados associados ao usuário.

Obter relatórios de credenciais (console)

Você pode usar o Console de gerenciamento da AWS para fazer download de um relatório de credenciais como um arquivo de valores separados por vírgula (CSV).

Para fazer download de um relatório de credenciais (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Relatório de credenciais.
3. Escolha Download Report (Fazer download do relatório).

Obter relatórios de credenciais (AWS CLI)

Para fazer download um relatório de credenciais (AWS CLI)

1. Gere um relatório de credenciais. O AWS armazena um único relatório. Se um relatório existir, a geração de um relatório de credenciais substituirá o relatório anterior. `aws iam generate-credential-report`
2. Exibir o último relatório gerado: `aws iam get-credential-report`

Obter relatórios de credenciais (API da AWS)

Para fazer download de um relatório de credenciais (API AWS)

1. Gere um relatório de credenciais. O AWS armazena um único relatório. Se um relatório existir, a geração de um relatório de credenciais substituirá o relatório anterior. `GenerateCredentialReport`
2. Exibir o último relatório gerado: `GetCredentialReport`

Credenciais do IAM para o CodeCommit: credenciais do Git, chaves SSH e chaves de acesso da AWS

O CodeCommit é um serviço de controle de versão gerenciado que hospeda repositórios privados do Git na Nuvem AWS. Para usar o CodeCommit, configure seu cliente Git para se comunicar com repositórios do CodeCommit. Como parte dessa configuração, forneça credenciais do IAM que podem ser usadas pelo CodeCommit para autenticar você. O IAM oferece suporte ao CodeCommit com três tipos de credenciais:

- Credenciais do Git, um par de nome de usuário e senha gerado pelo IAM que você pode usar para se comunicar com repositórios do CodeCommit por HTTPS.
- As chaves SSH, um par de chaves privada e pública gerado localmente, que você pode associar ao seu usuário do IAM para se comunicar com repositórios do CodeCommit por SSH.
- Chaves de acesso da AWS que você pode usar com o auxiliar de credenciais incluído na AWS CLI para se comunicar com repositórios do CodeCommit por HTTPS.

> **Note**
>
> Você não pode usar as chaves SSH ou as credenciais do Git para acessar repositórios em outra conta da AWS. Para saber como configurar o acesso aos repositórios do CodeCommit para usuários e grupos do IAM em outra Conta da AWS, consulte Configurar o acesso entre contas a um repositório do AWS CodeCommit usando perfis, no Guia do usuário do AWS CodeCommit.

Consulte as seções a seguir para obter mais informações sobre cada opção.

Usar as credenciais do Git e HTTPS com o CodeCommit (recomendado)

Com as credenciais do Git, é possível gerar um par de nome de usuário e senha estáticos para o seu usuário do IAM e, em seguida, usar essas credenciais para conexões HTTPS. Você também pode usar essas credenciais com qualquer ferramenta de terceiros ou ambiente de desenvolvimento integrado (IDE), que seja compatível com as credenciais estáticas do Git.

Como essas credenciais são universais para todos os sistemas operacionais com suporte e compatíveis com a maioria dos sistemas de gerenciamento de credenciais, ambientes de desenvolvimento e outras ferramentas de desenvolvimento de software, esse é o método recomendado. Você pode redefinir a senha para as credenciais do Git a qualquer momento. Você também pode tornar as credenciais inativas ou excluí-las se elas não forem mais necessárias.

> **Note**
>
> Você não pode escolher seu próprio nome do usuário ou senha para as credenciais do Git. O IAM gera essas credenciais para ajudar a garantir que atendam aos padrões de segurança da AWS e de repositórios seguros no CodeCommit. Você pode fazer download das credenciais somente uma vez, no momento em que elas são geradas. Certifique-se de salvar as credenciais em um local seguro. Se necessário, você pode redefinir a senha a qualquer momento, mas isso invalida todas as conexões configuradas com a senha antiga. Você deve reconfigurar as conexões para usar a nova senha antes de se conectar.

Consulte os tópicos a seguir para obter mais informações:

- Para criar um usuário do IAM, consulte Criar um usuário do IAM na Conta da AWS.

- Para gerar e usar as credenciais do Git com o CodeCommit, consulte Para usuários HTTPS que usam credenciais do Git no Guia do usuário do AWS CodeCommit.

> **Note**
>
> A alteração do nome de um usuário do IAM após a geração de credenciais do Git não altera o nome do usuário das credenciais do Git. O nome do usuário e a senha são os mesmos e ainda são válidos.

Para atualizar credenciais específicas do serviço

1. Crie um segundo conjunto de credenciais específicas do serviço além do conjunto em uso no momento.
2. Atualize todos os seus aplicativos para utilizar o novo conjunto de credenciais e confirme que os aplicativos estão funcionando.
3. Altere o estado das credenciais originais para "Inativas".
4. Certificar-se de que todos os seus aplicativos ainda estejam funcionando.
5. Exclua as credenciais inativas específicas do serviço.

Usar chaves SSH e SSH com o CodeCommit

Com conexões SSH, você cria arquivos de chave pública e privada em sua máquina local que o Git e o CodeCommit usam para a autenticação SSH. Associe a chave pública ao seu usuário do IAM e armazene a chave privada em sua máquina local. Consulte os tópicos a seguir para obter mais informações:

- Para criar um usuário do IAM, consulte Criar um usuário do IAM na Conta da AWS.
- Para criar uma chave pública SSH e associá-la a um usuário do IAM, consulte Para conexões SSH no Linux, macOS ou Unix ou consulte Para conexões SSH no Windows no Guia do usuário do AWS CodeCommit.

> **Note**
>
> A chave pública deve ser codificada no formato ssh-rsa ou PEM. O tamanho mínimo de bit da chave pública é de 2.048 bits, e o tamanho máximo é de 16.384 bits. Isso não tem relação

com o tamanho do arquivo que você carregou. Por exemplo, você pode gerar uma chave de 2.048 bits, e o arquivo PEM resultante terá 1.679 bytes. Se você fornecer a chave pública em outro formato ou tamanho, verá uma mensagem de erro indicando que o formato da chave é inválido.

Usar HTTPS com o auxiliar de credenciais da AWS CLI e o CodeCommit

Como uma alternativa às conexões HTTPS com credenciais do Git, você pode permitir que o Git use uma versão assinada com criptografia de suas credenciais de usuário do IAM ou uma função de instância do Amazon EC2 sempre que o Git precisar de autenticação na AWS para interagir com repositórios do CodeCommit. Este é o único método de conexão para repositórios do CodeCommit que não exige um usuário do IAM. Também é o único método que funciona com acesso federado e credenciais temporárias. Consulte os tópicos a seguir para obter mais informações:

- Para saber mais sobre o acesso federado, consulte Federação e provedores de identidade na AWS e Acesso aos usuários autenticados externamente (federação de identidades).
- Para saber mais sobre credenciais temporárias, consulte Credenciais de segurança temporárias no IAM e Acesso temporário a repositórios do CodeCommit.

O auxiliar de credenciais da AWS CLI não é compatível com outros sistemas auxiliares de credencial, como o Keychain Access ou o Windows Credential Management. Há considerações de configuração adicionais ao configurar conexões HTTPS com o auxiliar de credenciais. Para obter mais informações, consulte Para conexões HTTPS no Linux, macOS ou Unix com o auxiliar de credenciais da AWS CLI ou Conexões HTTPS no Windows com o auxiliar de credenciais da AWS CLI no Guia do usuário do AWS CodeCommit.

Gerenciar certificados de servidor no IAM

Para habilitar conexões HTTPS para o seu site ou aplicativo na AWS você precisa de um certificado de servidor SSL/TLS. Para certificados em uma região compatível com o Gerenciador de certificados da AWS (ACM), recomendamos que você use o ACM para provisionar, gerenciar e implantar seus certificados de servidor. Nas regiões sem suporte, você deve usar o IAM como gerenciador de certificados. Para saber quais regiões são compatíveis com o ACM, consulte Cotas e endpoints do Gerenciador de certificados da AWS na Referência geral da AWS.

> **Important**
>
> O ACM é a ferramenta preferencial para provisionar, gerenciar e implantar seus certificados de servidor. Com o ACM você pode solicitar um certificado ou implantar um ACM existente ou um certificado externo nos recursos da AWS. Os certificados fornecidos pelo ACM são gratuitos e são renovados automaticamente. Em uma região compatível, você pode usar o ACM para gerenciar certificados de servidor no console ou de forma programática. Para obter mais informações sobre o ACM, consulte o Guia do usuário do Gerenciador de certificados da AWS. Para obter mais informações sobre como solicitar um certificado do ACM, consulte Solicitar um certificado público ou Solicitar um certificado privado no Guia do usuário do Gerenciador de certificados da AWS. Para obter mais informações sobre a importação de certificados de terceiros para o ACM, consulte Importação de certificados no Manual do usuário do Gerenciador de certificados da AWS.

Use o IAM como um gerenciador de certificados apenas quando precisar oferecer suporte a conexões HTTPS em uma região que não é compatível com o ACM. O IAM criptografa com segurança suas chaves privadas e armazena a versão criptografada no armazenamento de certificado SSL do IAM. O IAM oferece suporte à implantação de certificados de servidor em todas as regiões, mas você deve obter seu certificado de um provedor externo para usar com a AWS. Você não pode carregar um certificado do ACM no IAM. Além disso, não é possível gerenciar seus certificados do console do IAM.

Para obter mais informações sobre como carregar certificados de terceiros para o IAM, consulte os tópicos a seguir.

Tópicos

- Fazer upload de um certificado de servidor (API da AWS)
- AWSOperações de API da para certificados de servidor
- Solucionar problemas com certificados de servidor

Fazer upload de um certificado de servidor (API da AWS)

Para carregar um certificado de servidor para o IAM, você deve fornecer o certificado e sua chave privada correspondente. quando o certificado não for autoassinado, você também deverá fornecer uma cadeia de certificado. (Você não precisa de uma cadeia de certificado ao fazer upload de um

certificado autoassinado.) Antes de fazer upload de um certificado, verifique se você tem todos estes itens e que atendem aos seguintes critérios:

- O certificado deve ser válido no momento do upload. Você não pode fazer upload de um certificado antes de seu período de validade começar (a data `NotBefore` do certificado) ou após sua expiração (a data `NotAfter` do certificado).
- A chave privada deve ser não criptografada. Você não pode fazer upload de uma chave privada que seja protegida por uma senha ou código de acesso. Para ajudar a descriptografar uma chave privada criptografada, consulte Solucionar problemas com certificados de servidor.
- O certificado, a chave privada e a cadeia de certificação devem ser codificados em PEM. Para ajudar a converter esses itens no formato PEM, consulte Solucionar problemas com certificados de servidor.

Para usar a API do IAM para carregar um certificado, envie uma solicitação UploadServerCertificate. O exemplo a seguir mostra como fazer isso com a AWS Command Line Interface (AWS CLI). O exemplo supõe o seguinte:

- O certificado codificado PEM está armazenado em um arquivo chamado `Certificate.pem`.
- A cadeia do certificado codificado PEM está armazenada em um arquivo chamado `CertificateChain.pem`.
- A chave privada não criptografada codificada PEM está armazenada em um arquivo chamado `PrivateKey.pem`.
- (Opcional) Você deseja etiquetar o certificado do servidor com um par de chave-valor. Por exemplo, você pode adicionar a chave de tag `Department` e o valor de tag `Engineering` para ajudar a identificar e organizar seus certificados.

Para usar o comando de exemplo a seguir, substitua os nomes de arquivo pelos seus próprios. Substitua *`ExampleCertificate`* pelo nome do seu certificado carregado. Se quiser marcar o certificado, substitua o par de chave-valor das tags *`ExampleKey`* e *`ExampleValue`* por seus próprios valores. Digite o comando em uma única linha contínua. O exemplo a seguir inclui quebras de linha e espaços extras para facilitar a leitura.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
```

```
                                    --tags '{"Key": "ExampleKey", "Value":
"ExampleValue"}'
```

Quando o comando anterior for executado com êxito, ele retornará metadados sobre o certificado carregado, incluindo nome de recurso da Amazon (ARN), nome fácil, identificador (ID), data de expiração, tags etc.

> **Note**
>
> Se você estiver carregando um certificado de servidor para usar com o Amazon CloudFront, deverá especificar um caminho usando a opção `--path`. O caminho deve começar com `/cloudfront` e deve incluir uma barra no final (por exemplo, `/cloudfront/test/`).

Para usar o AWS Tools for Windows PowerShell para fazer upload de um certificado, use Publish-IAMServerCertificate.

AWSOperações de API da para certificados de servidor

Use os comandos a seguir para visualizar, marcar, renomear e excluir certificados de servidor.

- Use GetServerCertificate para recuperar um certificado. Essa solicitação retorna o certificado, a cadeia de certificado (se foi carregado) e metadados sobre o certificado.

  > **Note**
  >
  > Você não pode baixar nem recuperar uma chave privada do IAM depois de carregá-lo.

- Use Get-IAMServerCertificate para recuperar um certificado.
- Use ListServerCertificates para listar seus certificados do servidor carregados. A solicitação retorna uma lista com metadados sobre cada certificado.
- Use Get-IAMServerCertificates para listar seus certificados de servidor carregados.
- Use TagServerCertificate para marcar um certificado de servidor existente.
- Use UntagServerCertificate para desmarcar um certificado de servidor.
- Use UpdateServerCertificate para renomear um certificado de servidor ou atualizar esse caminho.

  O exemplo a seguir mostra como fazer isso com a AWS CLI.

Para usar o exemplo de comando a seguir, substitua os nomes de certificados novo e antigo e o caminho do certificado e digite o comando em uma única linha contínua. O exemplo a seguir inclui quebras de linha e espaços extras para facilitar a leitura.

```
aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                  --new-server-certificate-
name CloudFrontCertificate
                                  --new-path /cloudfront/
```

Para usar o AWS Tools for Windows PowerShell para renomear um certificado de servidor ou atualizar seu caminho, use Update-IAMServerCertificate.

- Use DeleteServerCertificate para excluir um certificado de servidor.

  Para usar o AWS Tools for Windows PowerShell para excluir um certificado de servidor, use Remove-IAMServerCertificate.

Solucionar problemas com certificados de servidor

Antes de carregar um certificado para o IAM, você deve garantir que o certificado, a chave privada e a cadeia de certificados estejam todos codificados por PEM. Você também deve garantir que a chave privada não seja criptografada. Veja os exemplos a seguir.

Example Exemplo de certificado codificado em PEM

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Exemplo de chave privada não criptografada, codificada por PEM

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Exemplo de cadeia de certificado codificado em PEM

Uma cadeia de certificados contém um ou mais certificados. Você pode usar um editor de texto, o comando copy no Windows ou o comando cat do Linux para concatenar os arquivos de certificado

em uma cadeia. Quando você inclui vários certificados, cada certificado deve certificar o anterior. Você pode fazer isso concatenando os certificados, incluindo o certificado CA raiz por último.

O exemplo a seguir contém três certificados, mas sua cadeia de certificados pode conter mais ou menos certificados.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Se esses itens não estiverem no formato correto para carregamento no IAM, você poderá usar OpenSSL para convertê-los no formato correto.

Para converter um certificado ou uma cadeia de certificado de DER em PEM

Use o comando OpenSSL x509 como no exemplo a seguir. No exemplo a seguir, substitua o comando *Certificate.der* pelo nome do arquivo que contém o certificado codificado em DER. Substitua *Certificate.pem* pelo nome preferido do arquivo de saída para conter o certificado codificado por PEM.

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

Para converter uma chave privada de DER em PEM

Use o comando OpenSSL rsa como no exemplo a seguir. No exemplo a seguir, substitua o comando *PrivateKey.der* pelo nome do arquivo que contém sua chave privada codificada por DER. Substitua *PrivateKey.pem* pelo nome preferido do arquivo de saída para conter a chave privada codificada por PEM.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

Para descriptografar uma chave privada criptografada (remover a senha ou a frase secreta)

Use o comando OpenSSL rsa como no exemplo a seguir. Para usar o exemplo de comando a seguir, substitua *EncryptedPrivateKey.pem* pelo nome do arquivo que contém sua chave privada criptografada. Substitua *PrivateKey.pem* pelo nome preferido do arquivo de saída para conter a chave privada descriptografada codificada por PEM.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

Para converter um pacote de certificado de PKCS#12 (PFX) em PEM

Use o comando OpenSSL pkcs12 como no exemplo a seguir. No exemplo a seguir, substitua o comando *CertificateBundle.p12* pelo nome do arquivo que contém o pacote de certificado codificado por PKCS#12. Substitua *CertificateBundle.pem* pelo nome preferido do arquivo de saída para conter o pacote de certificado codificado por PEM.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

Para converter um pacote de certificado de PKCS#7 em PEM

Use o comando OpenSSL pkcs7 como no exemplo a seguir. No exemplo a seguir, substitua o comando *CertificateBundle.p7b* pelo nome do arquivo que contém o pacote de certificado codificado por PKCS#7. Substitua *CertificateBundle.pem* pelo nome preferido do arquivo de saída para conter o pacote de certificado codificado por PEM.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

Grupos de usuários do IAM

Um grupo de usuários do IAM é um conjunto de usuários do IAM. Os grupos de usuários permitem especificar permissões para vários usuários, o que pode facilitar o gerenciamento das permissões para esses usuários. Por exemplo, você pode ter um grupo de usuários chamado Admins e dar a esse grupo de usuários as permissões normais dos administradores. Qualquer usuário desse grupo de usuários tem automaticamente permissões de grupo Admins. Se um novo usuário ingressar

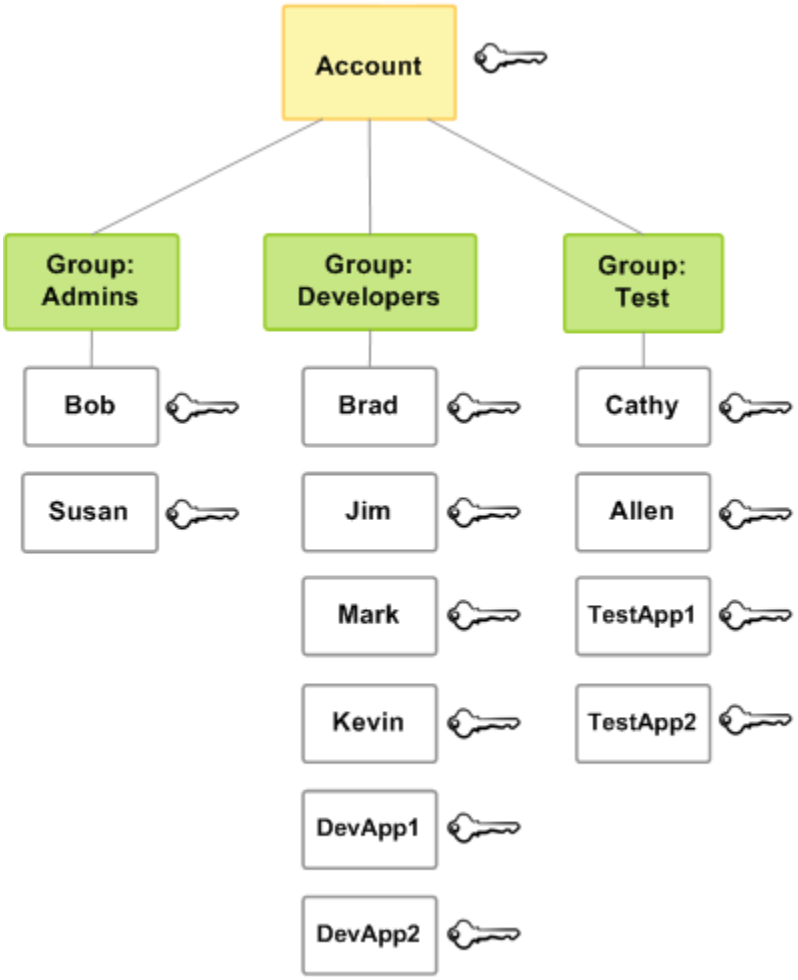
na organização e precisar de privilégios de administrador, você poderá atribuir as permissões apropriadas adicionando o usuário ao grupo de usuários Admins. Se alguém mudar de cargo na organização, em vez de editar as permissões desse usuário, você poderá remover a pessoa dos grupos do IAM antigos e adicioná-la aos novos grupos apropriados do IAM.

Você pode anexar uma política baseada em identidade a um grupo de usuários para que todos os usuários do grupo de usuários recebem as permissões da política. Não é possível identificar um grupo de usuários como `Principal` em uma política (como uma política baseada em recursos) porque os grupos são relacionados com permissões, não autenticação, e as entidades principais são entidades autenticadas do IAM. Para obter mais informações sobre tipos de política, consulte Políticas baseadas em identidade e em recurso.

Estas são algumas características importantes dos grupos do IAM:

- Um grupo de usuários pode conter muitos usuários e um usuário pode pertencer a vários grupos de usuários.
- Os grupos de usuários não podem ser aninhados e só podem conter usuários, não outros grupos do IAM.
- Não existe um grupo de usuários padrão que inclua automaticamente todos os usuários da Conta da AWS. Se você deseja ter um grupo de usuários como este, deve criá-lo e atribuir cada novo usuário a ele.
- O número e o tamanho dos recursos do IAM em uma Conta da AWS, como o número de grupos e o número de grupos dos quais um usuário pode ser membro, são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

O diagrama a seguir mostra um exemplo simples de uma pequena empresa. O proprietário da empresa cria um grupo de usuários `Admins` para que os usuários criem e gerenciem outros usuários à medida que a empresa cresce. O grupo de usuários `Admins` cria um grupo de usuários `Developers` e um grupo de usuários `Test`. Cada um desses grupos do IAM é composto de usuários (pessoas e aplicações) que interagem com a AWS (João, Brás, DevApp1, etc.). Cada usuário tem um conjunto individual de credenciais de segurança. Neste exemplo, cada usuário pertence a um único grupo de usuários. No entanto, os usuários podem pertencer a vários grupos do IAM.



Depois de criar um grupo de usuários, você pode Visualizar grupos do IAM, Anexar uma política a um grupo de usuários do IAM e Renomear um grupo de usuários do IAM.

Criar grupos do IAM

> **Note**
>
> Como prática recomendada, aconselhamos exigir que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias. Seguindo as práticas recomendadas, você não gerenciará usuários e grupos do IAM. Em vez disso, seus usuários e grupos serão gerenciados fora da AWS e podem acessar recursos da AWS como identidade federada. Identidade federada é um usuário de seu diretório de usuários corporativos, um provedor de identidades da Web, AWS Directory Service, o diretório do Centro de Identidade ou qualquer usuário que acesse os serviços da

AWS usando credenciais fornecidas por meio de uma fonte de identidade. As identidades federadas utilizam os grupos definidos pelo provedor de identidade. Se você estiver usando o Centro de Identidade do AWS IAM, consulte Manage identities in IAM Identity Center (Gerenciar identidades no Centro de Identidade do IAM) no Guia do usuário do Centro de Identidade do AWS IAM para obter informações sobre a criação de usuários e grupos no Centro de Identidade do IAM.

Os grupos do IAM são criados para ajudar gerenciar as permissões de acesso de vários usuários com perfis ou responsabilidades semelhantes. Ao anexar políticas a esses grupos, você pode conceder ou revogar permissões para conjuntos inteiros de usuários. Isso simplifica a manutenção das políticas de segurança, pois as alterações feitas nas permissões de um grupo são aplicadas automaticamente a todos os membros desse grupo, garantindo um controle de acesso consistente. Depois de criar o grupo, conceda permissões ao grupo com base no tipo de trabalho que você espera que os usuários do IAM no grupo façam e, em seguida, adicione os usuários do IAM ao grupo.

Para obter informações sobre as permissões necessárias para criar um grupo do IAM, consulte Permissões necessárias para acessar recursos do IAM.

Para criar um grupo do IAM e anexar políticas

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usuários) e escolha Create group (Criar grupo).
3. Em User group name (Nome do grupo de usuários), digite o nome do grupo.

> **Note**
>
> O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS. Os nomes dos grupos podem ser uma combinação de até 128 letras, dígitos e estes caracteres: mais (+), igual (=), vírgula (,), ponto (.), arroba (@), sublinhado (_) e hífen (-). Os nomes devem ser exclusivos dentro de uma conta. Não há diferenciação entre

maiúsculas e minúsculas. Por exemplo, não é possível criar grupos chamados de **`ADMINS`** e **`admins`**.

4. Na lista de usuários, marque a caixa de seleção para cada usuário que você deseja adicionar ao grupo.
5. Na lista de políticas, marque a caixa de seleção para cada política que você deseja aplicar a todos os membros do grupo.
6. Escolha Criar grupo.

AWS CLI

Execute o seguinte comando:

- aws iam create-group

API

Chame a seguinte operação:

- CreateGroup

Visualizar grupos do IAM

Você pode listar todos os grupos do IAM de sua conta, listar os usuários de um grupo de usuários e listar os grupos do IAM aos quais um usuário pertence. Se você usar a CLI ou a API, poderá listar todos os grupos do IAM com um determinado prefixo de caminho.

Console

Para listar todos os grupos do IAM na sua conta:

- No painel de navegação, selecione Grupos de usuários.

Para listar os usuários do IAM em um grupo do IAM específico:

- No painel de navegação, selecione User groups (Grupos de usuários). Escolha o nome do grupo de destino para abrir a página de detalhes do grupo. Revise a guia Usuários para ver a associação a grupos.

Para listar todos os grupos do IAM aos quais um usuário pertence:

- No painel de navegação, escolha Users. Em seguida, escolha o nome do usuário para abrir a página de detalhes do usuário. Escolha a guia Grupos para ver uma lista dos grupos aos quais o usuário pertence.

AWS CLI

Para listar todos os grupos do IAM na sua conta:

- aws iam list-groups

Para listar os usuários em um grupo do IAM específico:

- aws iam get-group

Para listar todos os grupos do IAM aos quais um usuário pertence:

- aws iam list-groups-for-user

API

Para listar todos os grupos do IAM na sua conta:

- ListGroups

Para listar os usuários em um grupo do IAM específico:

- GetGroup

Para listar todos os grupos do IAM aos quais um usuário pertence:

- ListGroupsForUser

Editar usuários em grupos do IAM

Use os grupos do IAM para aplicar as mesmas políticas de permissões a vários usuários de uma só vez. É possível adicionar ou remover usuários de um grupo do IAM. Isso é útil à medida que pessoas entram e saem de sua organização.

Revisar acesso à política

Antes de remover um grupo, use a página de detalhes do grupo para analisar os membros (usuários do IAM) do grupo, as políticas anexadas ao grupo na guia Permissões e revisar a atividade recente em nível de serviço usando a guia Último acesso. Isso ajuda a evitar a remoção não intencional do acesso de uma entidade principal (pessoa ou aplicação) que o esteja usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Adicionar um usuário do IAM a um grupo do IAM

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usuários) e, em seguida, escolha o nome do grupo.
3. Escolha a guia Users (Usuários) e, em seguida, Add users (Adicionar usuários). Marque a caixa de seleção próxima dos usuários que você deseja adicionar.
4. Escolha Add users (Adicionar usuários).

AWS CLI

Execute o seguinte comando:

- `aws iam add-user-to-group`

API

Chame a seguinte operação:

- `AddUserToGroup`

Remover um usuário do IAM de um grupo do IAM

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usuários) e, em seguida, escolha o nome do grupo.
3. Escolha a guia Users. Marque a caixa de seleção ao lado dos usuários que deseja remover e escolha Remove users (Remover usuários).

AWS CLI

Execute o seguinte comando:

- `aws iam remove-user-from-group`

API

Chame a seguinte operação:

- `RemoveUserFromGroup`

Anexar uma política a um grupo de usuários do IAM

Você pode anexar uma política gerenciada pela AWS, ou seja, uma política pré-escrita fornecida pela AWS, a um grupo de usuários, conforme explicado nas etapas a seguir. Para anexar uma política gerenciada pelo cliente, ou seja, uma política com permissões personalizadas criada por você, crie a política primeiro. Para obter informações sobre a criação de políticas gerenciadas pelo cliente, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.

Para obter informações sobre permissões e políticas, consulte Gerenciamento de acesso para recursos da AWS.

Para anexar uma política a um grupo do IAM

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usuários) e, em seguida, escolha o nome do grupo.
3. Escolha a aba Permissões.
4. Escolha Adicionar permissões e depois Anexar políticas.
5. As políticas atuais anexadas ao grupo de usuários são exibidas na lista Current permissions policies (Políticas de permissões atuais). Na lista Other permissions policies (Outras políticas de permissões), marque a caixa de seleção ao lado dos nomes das políticas a serem anexadas. Você pode usar a caixa de pesquisa para filtrar a lista de políticas por tipo e nome de política.
6. Selecione a política que deseja anexar ao seu grupo do IAM e escolha Anexar políticas.

AWS CLI

Execute o seguinte comando:

- `aws iam attach-group-policy`

API

Chame a seguinte operação:

- `AttachGroupPolicy`

Renomear um grupo de usuários do IAM

Quando você altera o nome ou o caminho de um grupo de usuários, acontece o seguinte:

- Todas as políticas anexadas ao grupo de usuários permanecem com o grupo sob o novo nome.
- O grupo de usuários retém todos os seus usuários com o novo nome.
- O ID exclusivo do grupo de usuários permanece o mesmo. Para obter mais informações sobre IDs exclusivos, consulte Identificadores exclusivos.

O IAM não atualiza automaticamente as políticas que se referem ao grupo de usuários como um recurso para usar o novo nome. Portanto, você deve ter cuidado ao renomear um grupo de usuários. Antes de renomear o grupo de usuários, você deve verificar manualmente todas as suas políticas para encontrar as políticas em que esse grupo de usuários é mencionado pelo nome. Por exemplo, digamos que Bob seja gerente da parte de testes da organização. Bob tem uma política anexada à entidade de usuário do IAM que permite a ele adicionar e remover usuários do grupo de usuários Teste. Se um administrador alterar o nome do grupo de usuários (ou mudar o caminho do grupo), ele também precisará atualizar a política anexada a Bob para usar o novo nome ou o novo caminho. Caso contrário, Bob não poderá adicionar e remover usuários do grupo de usuários.

Para encontrar as políticas que se referem a um grupo do IAM como um recurso:

1. No painel de navegação do console do IAM, escolha Policies (Políticas).
2. Classifique pela coluna Type (Tipo) para encontrar suas políticas personalizadas Customer managed (Gerenciadas pelo cliente).
3. Escolha o nome da política a ser editada.
4. Escolha a guia Permissions e escolha Resumo.
5. Escolha IAM na lista de serviços, se houver esta opção.
6. Procure o nome do seu grupo de usuários na coluna Resource (Recurso).
7. Escolha Editar para alterar o nome do grupo de usuários na política.

Para alterar o nome de um grupo de usuários do IAM

Console

1. No painel de navegação, escolha Grupos de usuários e, em seguida, selecione o nome do grupo.
2. Escolha Editar. Digite o novo nome do grupo de usuários e escolha Save changes (Salvar alterações).

AWS CLI

Execute o seguinte comando:

- aws iam update-group

API

Chame a seguinte operação:

- UpdateGroup

Excluir um grupo do IAM

Quando um grupo do IAM é excluído no console, o console remove automaticamente todos os membros do grupo, desanexa todas as políticas gerenciadas e exclui todas as políticas em linha. Contudo, como o IAM não exclui automaticamente as políticas que se referem ao grupo do IAM como um recurso, você deve tomar cuidado ao excluir um grupo do IAM. Antes de excluir seu grupo do IAM, revise manualmente as suas políticas para encontrar quaisquer políticas que mencionem o grupo por nome. Por exemplo, João tem uma política anexada à sua entidade de usuário do IAM que permite que ele adicione e remova usuários do grupo de usuários Teste. Se um administrador excluir o grupo, ele também precisará excluir a política anexada a John. Caso contrário, se o administrador recriar o grupo excluído e dar a ele o mesmo nome, as permissões de João permanecerão em vigor, mesmo que ele tenha saído da Equipe de teste.

Por outro lado, ao usar a CLI, o SDK ou a API para excluir um grupo, primeiro é necessário remover os usuários do grupo. Exclua então todas as políticas em linha incorporadas no grupo do IAM. Em seguida, desanexe todas as políticas gerenciadas anexadas ao grupo. Agora exclua o grupo do IAM propriamente dito.

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione User groups (Grupos de usuários).
3. Na lista de grupos do IAM, marque a caixa de seleção ao lado dos nomes dos grupos do IAM a serem excluídos. É possível usar a caixa de pesquisa para filtrar a lista de grupos do IAM por tipo, permissões e nome do grupo.
4. Selecione a opção Excluir.
5. Na caixa de confirmação, para excluir um único grupo, digite o nome do grupo e escolha Excluir. Se você quiser excluir vários grupos, digite o número de grupos do IAM a serem excluídos, seguido por **`user groups`**, e escolha Excluir. Por exemplo, para excluir três grupos do IAM, digite **`3 user groups`**.

AWS CLI

1. Remova todos os usuários do grupo do IAM.
   - aws iam get-group (para obter a lista de usuários no grupo do IAM) e aws iam remover-user-from-group (para remover um usuário do grupo do IAM)
2. Exclua todas as políticas em linha incorporadas no grupo do IAM.
   - aws iam list-group-policies (para obter uma lista de políticas em linha do grupo do IAM) e aws iam delete-group-policy (para excluir as políticas em linha do grupo do IAM)
3. Desanexe todas as políticas gerenciadas anexadas ao grupo do IAM.
   - aws iam list-attached-group-policies (para obter uma lista das políticas gerenciadas anexados ao grupo do IAM) e aws iam detach-group-policy (para desanexar uma política gerenciada do grupo do IAM)
4. Exclua o grupo do IAM.
   - aws iam delete-group

API

1. Remova todos os usuários do grupo do IAM.
   - GetGroup (para obter a lista de usuários no grupo do IAM) e RemoveUserFromGroup (para remover um usuário do grupo do IAM)
2. Exclua todas as políticas em linha incorporadas no grupo do IAM.
   - ListGroupPolicies (para obter uma lista de políticas em linha do grupo do IAM) e DeleteGroupPolicy (para excluir as políticas em linha do grupo do IAM)
3. Desanexe todas as políticas gerenciadas anexadas ao grupo do IAM.
   - ListAttachedGroupPolicies (para obter uma lista das políticas gerenciadas anexados ao grupo do IAM) e DetachGroupPolicy (para desanexar uma política gerenciada do grupo do IAM)
4. Exclua o grupo do IAM.
   - DeleteGroup

Perfis do IAM

Uma função do IAM é uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. Uma função do IAM é semelhante a um usuário do IAM no sentido de que é uma identidade da AWS com políticas de permissão que determinam o que a identidade pode e não pode fazer na AWS. No entanto, em vez de ser exclusivamente associada a uma pessoa, o propósito do perfil é ser assumido por qualquer pessoa que precisar dele. Além disso, um perfil não tem credenciais de longo prazo padrão associadas a ele, como senha ou chaves de acesso. Em vez disso, quando você assumir um perfil, ele fornecerá credenciais de segurança temporárias para sua sessão de perfil.

É possível usar funções para delegar acesso a usuários, aplicações ou serviços que normalmente não têm acesso aos seus recursos da AWS. Por exemplo, você pode conceder para os usuários na sua conta da AWS acesso a recursos que normalmente eles não têm ou conceder para os usuários em uma Conta da AWS acesso a recursos em outra conta. Também é possível permitir que uma aplicação móvel use recursos da AWS, mas sem incorporar chaves da AWS na aplicação (onde pode ser difícil atualizá-las e onde os usuários poderão potencialmente extraí-las). Às vezes, você deseja oferecer acesso à AWS a usuários que já têm identidades definidas fora da AWS, como no diretório corporativo. Você também pode conceder acesso à sua conta a terceiros, para que eles possam realizar uma auditoria em seus recursos.

Para esses cenários, você pode delegar acesso aos recursos da AWS usando uma função do IAM. Esta seção apresenta funções e as diferentes maneiras de usá-las, quando e como escolher essas abordagens e como criar, gerenciar, alternar para (ou assumir) e excluir funções.

O gerenciador de acesso à conta é um atributo do IAM que permite atribuir centralmente os perfis do IAM nas contas da sua organização aos usuários e grupos do IAM Identity Center. O gerenciador de acesso à conta concede o acesso à força de trabalho por meio de perfis do IAM com o respectivo conjunto completo de atributos. Você pode usá-lo junto com os conjuntos de permissões do IAM Identity Center ou sozinho.

> **Note**
>
> Quando você cria sua Conta da AWS, nenhum perfil é criado por padrão. Conforme você adiciona serviços à sua conta, eles podem adicionar perfis vinculados a serviços para dar suporte a seus casos de uso.
> Um perfil vinculado a serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode presumir o perfil de executar uma ação em seu nome.

Perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.
Para poder excluir esses perfis vinculados a serviços, você deve primeiro excluir os recursos relacionados a eles. Isso protege seus recursos porque você não pode remover por engano as permissões para acessar os recursos.
Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Tópicos

- Quando criar um usuário do IAM (em vez de uma função)
- Termos e conceitos das funções
- Recursos adicionais
- O problema "confused deputy"
- Cenários comuns para perfis do IAM
- Criar um perfil do IAM
- Gerenciamento de perfis do IAM
- Métodos para assumir um perfil

Quando criar um usuário do IAM (em vez de uma função)

Recomendamos usar somente usuários do IAM para casos de uso sem suporte à federação de identidades. Alguns dos casos de uso incluem o seguinte:

- Workloads que não podem usar perfis do IAM: você pode executar a workload de um local que precisa acessar a AWS. Em algumas situações, você não pode usar perfis do IAM para fornecer credenciais temporárias, como para plugins do WordPress. Nessas situações, use as chaves de acesso de longo prazo do usuário do IAM para que essa workload seja autenticada na AWS.
- Clientes de terceiros da AWS: se você estiver usando ferramentas que não oferecem suporte ao acesso com o Centro de Identidade do IAM, como fornecedores ou clientes terceiros da AWS que não estão hospedados na AWS, use as chaves de acesso de longo prazo do usuário do IAM.

- Acesso ao AWS CodeCommit: se você estiver usando o CodeCommit para armazenar seu código, poderá usar um usuário do IAM com chaves SSH ou credenciais específicas de serviço para que o CodeCommit seja autenticado em seus repositórios. Recomendamos fazer isso além de usar um usuário do IAM Identity Center para autenticação padrão. Os usuários do Centro de Identidade do IAM são as pessoas em sua força de trabalho que precisam acessar suas Contas da AWS ou suas aplicações na nuvem. Para dar aos usuários acesso aos seus repositórios do CodeCommit sem configurar os usuários do IAM, você pode configurar o utilitário git-remote-codecommit. Para obter mais informações sobre o IAM e o CodeCommit, consulte Credenciais do IAM para o CodeCommit: credenciais do Git, chaves SSH e chaves de acesso da AWS. Para obter mais informações sobre como configurar o utilitário git-remote-codecommit, consulte Conectar-se a repositórios do AWS CodeCommit credenciais alternadas, no Guia do usuário do AWS CodeCommit.
- Acesso ao Amazon Keyspaces (para Apache Cassandra): em uma situação em que não é possível usar usuários no IAM Identity Center, como para fins de teste de compatibilidade com o Cassandra, você pode usar um usuário do IAM com credenciais específicas do serviço para a autenticação com o Amazon Keyspaces. Os usuários do Centro de Identidade do IAM são as pessoas em sua força de trabalho que precisam acessar suas Contas da AWS ou suas aplicações na nuvem. Você também pode se conectar ao Amazon Keyspaces usando credenciais temporárias. Para obter mais informações, consulte Using temporary credentials to connect to Amazon Keyspaces using an IAM role and the SigV4 plugin (Como usar credenciais temporárias para se conectar ao Amazon Keyspaces usando um perfil do IAM e o plug-in SigV4) no Guia do desenvolvedor do Amazon Keyspaces (para Apache Cassandra).
- Acesso de emergência: em uma situação em que você não consegue acessar seu provedor de identidade e precisa tomar medidas em sua Conta da AWS. Estabelecer usuários do IAM com acesso emergencial pode fazer parte do plano de resiliência. Recomendamos que as credenciais do usuário de emergência sejam rigidamente controladas e protegidas por autenticação multifator (MFA).

Termos e conceitos das funções

Veja a seguir alguns termos básicos para começar a usar as funções.

Perfil

Uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. Uma função do IAM tem algumas semelhanças com um usuário do IAM. Funções e usuários são identidades da AWS com políticas de permissão que determinam o que a identidade pode

e não pode fazer na AWS. No entanto, em vez de ser exclusivamente associada a uma pessoa, o propósito do perfil é ser assumido por qualquer pessoa que precisar dele. Além disso, um perfil não tem credenciais de longo prazo padrão associadas a ele, como senha ou chaves de acesso. Em vez disso, quando você assumir um perfil, ele fornecerá credenciais de segurança temporárias para sua sessão de perfil.

Os perfis podem ser usados pelas seguintes entidades:

- Um usuário do IAM na mesma Conta da AWS ou em outra Conta da AWS
- Perfis do IAM na mesma conta
- Entidades principais de serviço, para uso com serviços e recursos da AWS como:
  - Serviços que permitem executar código em serviços de computação, como Amazon EC2 ou AWS Lambda
  - Atributos que executam ações em seus recursos em seu nome, como a replicação de objetos do Amazon S3
  - Serviços que fornecem credenciais de segurança temporárias para suas aplicações executadas fora da AWS, como IAM Roles Anywhere ou Amazon ECS Anywhere
- Um usuário externo autenticado por um serviço de provedor de identidades (IdP) compatível com SAML 2.0 ou OpenID Connect

AWS Perfil de serviço da

O perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criação de um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

AWS Função vinculada ao serviço do

Um perfil vinculado a serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode presumir o perfil de executar uma ação em seu nome. Perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.

> **Note**
>
> Se já estiver usando um serviço quando ele começar a oferecer suporte às funções vinculadas ao serviço, você poderá receber um e-mail informando sobre uma nova função na sua conta. Nesse caso, o serviço cria automaticamente a função vinculada ao serviço na sua conta. Você não precisa realizar nenhuma ação para oferecer suporte a essa

função, e você não deve excluir manualmente. Para obter mais informações, consulte Uma nova função apareceu na minha conta da AWS.

Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço. Para obter mais informações, consulte Criar um perfil vinculado ao serviço.

Encadeamento de funções

O encadeamento de perfis ocorre quando um perfil é usado para assumir um segundo perfil. Você pode realizar o encadeamento de perfis via Console de gerenciamento da AWS alternando entre os perfis, a AWS CLI ou a API. Por exemplo, suponha que o `RoleA` tenha permissão para assumir o `RoleB`. É possível permitir que o Usuário1 assuma o `RoleA` usando as credenciais de usuário de longo prazo na operação da API AssumeRole. Esta operação retorna as credenciais de curto prazo do `RoleA`. Para iniciar o encadeamento de funções, você pode usar as credenciais de curto prazo do `RoleA` para permitir que o Usuário1 assuma o `RoleB`.

Ao assumir uma função, você pode passar uma tag de sessão e definir a tag como transitiva. As tags de sessão transitivas são passadas para todas as sessões subsequentes em uma cadeia de funções. Para saber mais sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

O encadeamento de perfis limita a sessão de perfil da API do Console de gerenciamento da AWS, AWS CLI ou AWS a um máximo de uma hora. Isso se aplica de forma independente da duração máxima da sessão configurada para perfis individuais. Ao usar a operação de API AssumeRole para assumir uma função, você pode especificar o tempo da sessão da sua função usando o parâmetro `DurationSeconds`. É possível especificar um valor de parâmetro de até 43200 segundos (12 horas), dependendo da configuração da duração máxima da sessão para seu perfil. No entanto, se você assumir uma função usando o encadeamento e fornecer um valor de parâmetro `DurationSeconds` maior do que uma hora, a operação falhará.

Para obter mais informações sobre como alternar entre perfis no Console de gerenciamento da AWS, consulte Mudar de um usuário para um perfil do IAM (console).

Delegação

A concessão de permissões a alguém para permitir o acesso a recursos controlados por você. A delegação envolve estabelecer confiança entre duas contas. A primeira é a conta que possui

o recurso (a conta de confiança). A segunda é a conta que contém os usuários que precisam acessar o recurso (a conta confiável). As contas confiável e de confiança podem ser qualquer uma das seguintes:

- A mesma conta.
- Contas separadas que estão sob controle da sua organização.
- Duas contas pertencentes a organizações diferentes.

Para delegar permissão para acessar um recurso, você cria um perfil do IAM na conta de confiança com duas políticas anexadas. A política de permissões concede ao usuário da função as permissões necessárias para executar as tarefas pretendidas no recurso. A política de confiança especifica quais membros das contas confiáveis têm permissão para assumir a função.

Ao criar uma política de confiança, você não pode especificar um curinga (*) como parte de um ARN no elemento de entidade principal. A política de confiança é anexada à função na conta de confiança e equivale à metade das permissões. A outra metade é uma política de permissões anexada ao usuário na conta confiável que permite a ele alternar ou assumir a função. Um usuário que assume uma função temporariamente desiste de suas próprias permissões e, em vez disso, assume as permissões da função. Quando o usuário sai ou para de usar a função, as permissões originais do usuário são restauradas. Um parâmetro adicional chamado ID externo ajuda a garantir o uso seguro de funções entre contas que não são controladas pela mesma organização.

Política de confiança

Um documento de política JSON no qual você define os principais nos quais você confia para assumir a função. Uma política de confiança da função é uma política com base em recurso necessária anexada a uma função no IAM. Os principais que podem ser especificados na política de confiança incluem usuários, funções, contas e serviços. Para obter mais informações, consulte Como usar políticas de confiança em perfis do IAM no AWS Security Blog.

Função para acesso entre contas

Uma função que concede o acesso a recursos em uma conta a uma entidade principal confiável em outra conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, alguns dos serviços da AWS permitem que você anexe uma política diretamente a um recurso (em vez de usar uma função como proxy). Essas políticas são chamadas de políticas baseadas em recursos, e você pode usá-las para conceder acesso ao recurso para entidades principais em outra Conta da AWS. Alguns desses recursos incluem buckets do Amazon Simple Storage Service (S3), cofres do Amazon Glacier, tópicos do Amazon Simple Notification Service (SNS)

e filas do Amazon Simple Queue Service (SQS). Para saber quais serviços oferecem suporte a políticas baseadas em recursos, consulte AWSServiços da que funcionam com o IAM. Para obter mais informações sobre políticas baseadas em recursos, consulte Acesso a recursos entre contas no IAM.

Recursos adicionais

Os seguintes recursos podem ajudar você a saber mais sobre a terminologia do IAM relacionada a perfis do IAM.

- Entidades principais são entidades na AWS que podem executar ações e acessar recursos. Uma entidade principal pode ser um Usuário raiz da conta da AWS, um usuário do IAM ou um perfil. Uma entidade principal que representa a identidade de um serviço da AWS é uma entidade principal de serviço. Use o elemento Principal nas políticas de confiança do perfil para definir as entidades principais em que você confia para assumir o perfil.

  Para obter mais informações e exemplos de entidades principais que você pode permitir que assumam um perfil, consulte AWS Elementos da política JSON da: Principal.

- A federação de identidades cria uma relação de confiança entre um provedor de identidades externo e a AWS. É possível usar seu provedor de OpenID Connect (OIDC) ou Security Assertion Markup Language (SAML) 2.0 existente para gerenciar quem pode acessar os recursos da AWS. Quando você usa o OIDC e o SAML 2.0 para configurar uma relação de confiança entre esses provedores de identidades externos e a AWS, o usuário é atribuído a um perfil do IAM. O usuário também recebe credenciais temporárias que permitem que ele acesse seus recursos da AWS.

  Para obter mais informações sobre entidades principais federadas, consulte Federação e provedores de identidade na AWS.

- Entidades principais federadas são identidades existentes do Directory Service, do diretório de usuários da sua empresa ou de um provedor OIDC. A AWS atribui um perfil a uma entidade principal federada quando o acesso é solicitado por meio de um provedor de identidade.

  Para obter mais informações sobre as entidades principais federadas SAML e OIDC, consulte Sessões de usuários federados e perfis.

- As políticas de permissões são políticas baseadas em identidade que definem quais ações e recursos o perfil pode usar. O documento é criado de acordo com as regras da linguagem da política do IAM.

Para obter mais informações, consulte Referência de política JSON do IAM.

- Os limites de permissões são um recurso avançado em que você usa políticas para limitar as permissões máximas que uma política baseada em identidade pode conceder a uma função. Você não pode aplicar um limite de permissões a uma função vinculada ao serviço.

  Para obter mais informações, consulte Limites de permissões para entidades do IAM.

O problema "confused deputy"

O problema "confused deputy" é um problema de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executar a ação. Para evitar isso, a AWS fornece ferramentas que ajudam você a proteger sua conta se você fornecer acesso aos recursos na sua conta a terceiros (conhecido como entre contas) ou a outros serviços da AWS (conhecido como entre serviços).

Às vezes, poderá ser necessário conceder acesso aos recursos da AWS a terceiros (delegar acesso). Por exemplo, você decide contratar uma empresa terceirizada chamada Example Corp para monitorar sua Conta da AWS e ajudar a otimizar os custos. Para rastrear seus gastos diários, a Example Corp precisa de acesso aos seus recursos da AWS. A Example Corp também monitora muitas outras Contas da AWS para outros clientes. É possível usar um perfil do IAM para estabelecer uma relação de confiança entre sua Conta da AWS e a conta da Example Corp. Um aspecto importante desse cenário é o ID externo, um identificador opcional que você pode usar em uma política de confiança do perfil do IAM para designar quem pode assumir ao perfil. A função principal do ID externo é abordar e impedir o problema "confused deputy".

Alguns serviços (serviços de chamada) da AWS usam a entidade principal do serviço da AWS para acessar recursos da AWS de outros serviços da AWS (serviços chamados). Em algumas dessas interações de serviços, você pode configurar serviços de chamada para se comunicarem com os recursos de um serviço chamado em outra Conta da AWS. Um exemplo disso é configurar o AWS CloudTrail para gravar em um bucket central do Amazon S3 localizado em outra Conta da AWS. É concedido ao serviço de chamadas CloudTrail acesso ao bucket do S3 usando a política de bucket do S3 por meio da adição de uma instrução de permissão para `cloudtrail.amazonaws.com`.

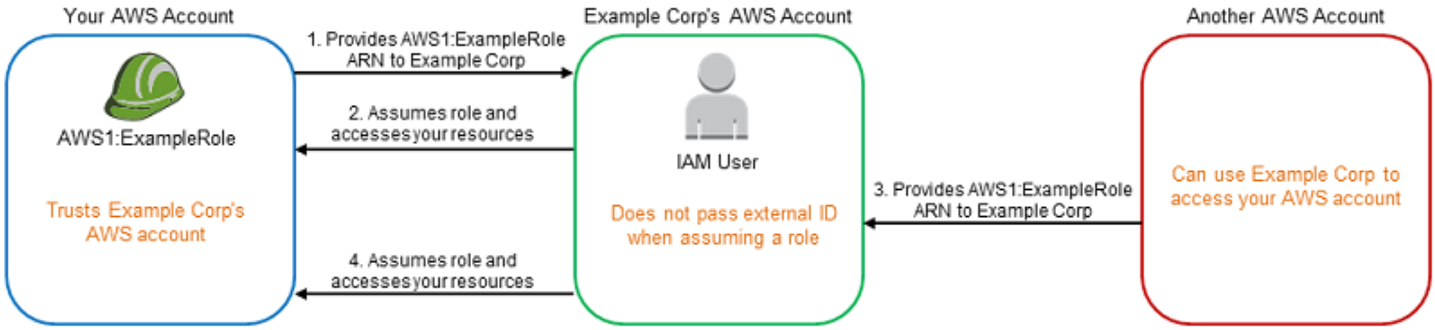
Quando uma entidade principal de serviço da AWS de um serviço de chamada está acessando um recurso de um serviço chamado, a política de recursos do serviço chamado está autorizando somente a entidade principal do serviço AWS, e não o ator que configurou o serviço de chamada. Por exemplo, um bucket do S3 que confia na entidade principal do serviço CloudTrail sem condições

pode receber logs do CloudTrail das Contas da AWS configuradas por um administrador confiável, mas também logs do CloudTrail de um agente não autorizado em sua Conta da AWS, se ele souber o nome do bucket do Amazon S3.

O problema "confused deputy" surge quando um ator usa a confiança de uma entidade principal de um serviço da AWS para obter acesso a recursos aos quais não deveria ter acesso.

Prevenção de "confused deputy" entre contas

O diagrama a seguir ilustra o problema "confused deputy" entre contas.



Este cenário supõe o seguinte:

- AWS1 é a sua Conta da AWS.
- AWS1:ExampleRole é uma função em sua conta. Esta política de confiança da função confia na Example Corp especificando a conta da AWS da Example Corp como a conta que pode assumir a função.

Veja o que acontece:

1. Ao começar a usar o serviço da Exemplo Corp, você fornece o ARN AWS1:ExampleRole à Exemplo Corp.
2. A Example Corp usa esse ARN de perfil para obter credenciais de segurança temporárias para acessar recursos na sua Conta da AWS. Dessa forma, confie na Example Corp como um "substituto" que pode agir em seu nome.
3. Outro cliente da AWS também começa a usar os serviços da Exemplo Corp e também fornece o ARN AWS1:ExampleRole para a Exemplo Corp usar. Provavelmente, o outro cliente soube ou adivinhou o AWS1:ExampleRole, que não é um segredo.

4. Quando o outro cliente pede à Exemplo Corp que acesse os recursos da AWS (o que ele afirma ser) em sua conta, a Exemplo Corp usa AWS1:ExampleRole para acessar os recursos da sua conta.

Dessa forma, o outro cliente pode obter acesso não autorizado aos seus recursos. Como esse outro cliente foi capaz de burlar a Example Corp para agir involuntariamente em seus recursos, a Example Corp agora é um "confused deputy."

A Example Corp pode abordar o problema "confused deputy" exigindo que você inclua a verificação de condição `ExternalId` na política de confiança da função. A Example Corp gera um único valor de `ExternalId` para cada cliente e usa esse valor em sua solicitação para assumir a função. O valor de `ExternalId` deve ser exclusivo entre os clientes da Example Corp e controlado pela Example Corp, não por seus clientes. É por isso que ele é fornecido pela Example Corp e não é criado por você. Isso impede que a Example Corp seja um "confused deputy" e conceda acesso a recursos da AWS de outra conta.

Em nosso cenário, imagine que seu identificador exclusivo da Example Corp seja 12345 e que o identificador do outro cliente seja 67890. Esses identificadores são simplificados para esse cenário. Em geral, esses identificadores são GUIDs. Supondo que esses identificadores sejam exclusivos entre os clientes da Example Corp, eles são valores confidenciais próprios para o ID externo.

A Example Corp atribui o valor do ID externo "12345" a você. É necessário adicionar um elemento `Condition` à política de confiança do perfil que exija que o valor `sts:ExternalId` seja 12345, da seguinte forma:

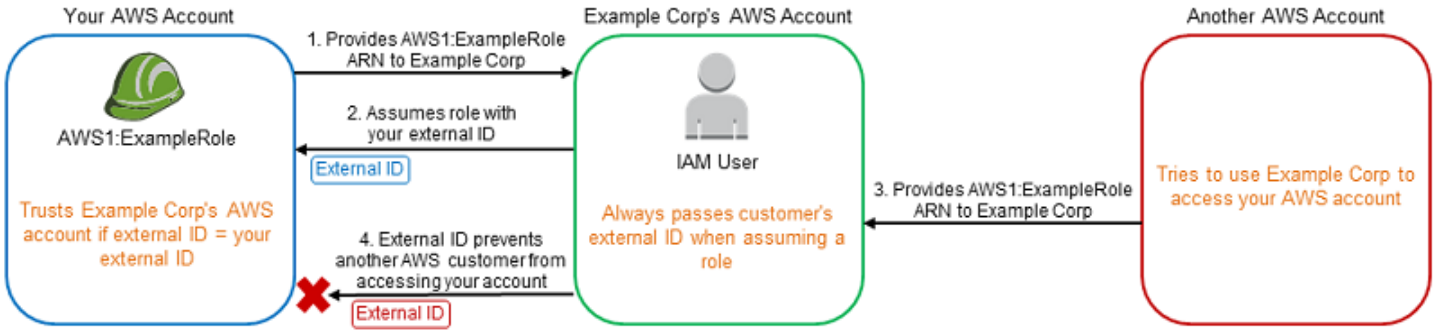
JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
```

```
      }
    }
}
```

O elemento Condition (Condição) dessa política só permite que a Example Corp assuma a função quando a chamada de API AssumeRole incluir o valor do ID externo 12345. A Example Corp garante que sempre que assumir uma função em nome de um cliente, incluirá o valor do ID externo desse cliente em uma chamada AssumeRole. Mesmo que outro cliente forneça seu ARN à Example Corp, ele não poderá controlar o ID externo que a Example Corp inclui em sua solicitação para a AWS. Isso ajuda a evitar que um cliente não autorizado tenha acesso aos seus recursos.

O diagrama a seguir ilustra isso.

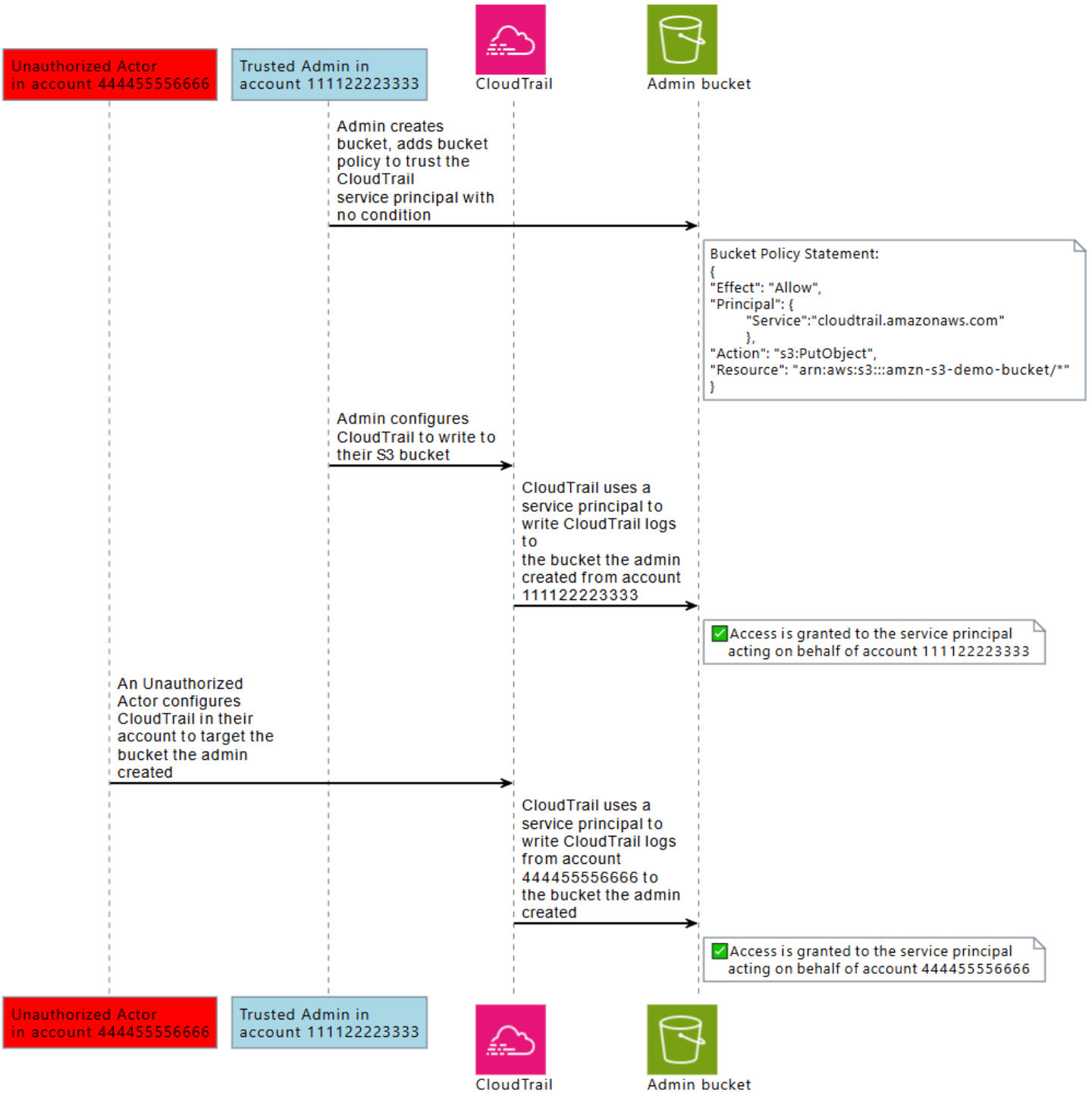


1. Como antes, ao começar a usar o serviço da Exemplo Corp, você fornece o ARN AWS1:ExampleRole à Exemplo Corp.
2. Quando a Exemplo Corp usa esse ARN de perfil para assumir a função AWS1:ExampleRole, ela inclui o ID externo (12345) na chamada de API AssumeRole. O ID externo corresponde à política de confiança da perfil e, portanto, a chamada de API AssumeRole tem êxito, e a Example Corp obtém credenciais de segurança temporárias para acessar recursos na sua Conta da AWS.
3. Outro cliente da AWS também começa a usar os serviços da Exemplo Corp e, assim como antes, esse cliente também fornece o ARN AWS1:ExampleRole para a Exemplo Corp usar.
4. Mas, desta vez, quando a Exemplo Corp tenta assumir a função AWS1:ExampleRole, ela fornece o ID externo associado a outro cliente (67890). O outro cliente não tem como alterar isso. A Example Corp faz isso porque a solicitação para usar a função veio de outro cliente, portanto, 67890 indica a circunstância em que a Example Corp está atuando. Como você adicionou uma condição com seu próprio ID externo (12345) à política de confiança de AWS1:ExampleRole, a chamada de API AssumeRole não vai funcionar. O outro cliente será impedido de obter o acesso não autorizado aos recursos na sua conta (indicado pelo "X" vermelho no diagrama).

O ID externo ajuda a prevenir que qualquer outro cliente engane a Example Corp a acessar seus recursos involuntariamente.

Prevenção do problema "confused deputy" entre serviços

O diagrama a seguir demonstra o problema "confused deputy" entre serviços usando o exemplo de interação entre o CloudTrail e o Amazon S3, em que um ator não autorizado grava logs do CloudTrail em um bucket do Amazon S3 ao qual não está autorizado a ter acesso.

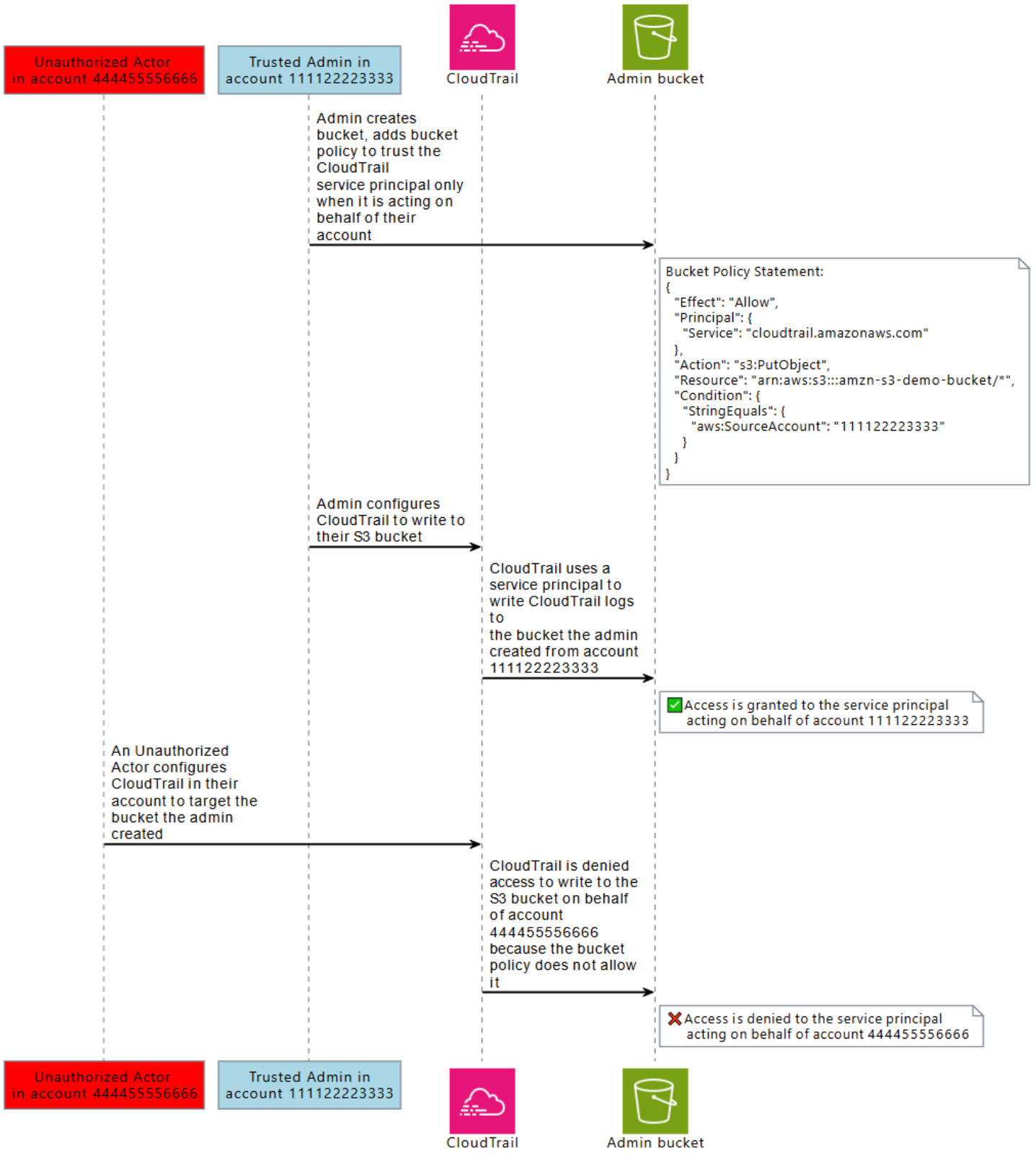


Para ajudar a evitar que um ator não autorizado use a confiança de uma entidade principal da AWS para obter acesso aos seus recursos, as entidades principais de serviços da AWS incluem informações sobre o recurso da AWS, a Conta da AWS e a organização da AWS em nome dos quais estão agindo.

Essas informações estão disponíveis em valores-chave de condições globais que podem ser usados em uma política de recursos ou política de controle de recursos para solicitações feitas pelas entidades principais de serviços da AWS. Recomendamos usar aws:SourceArn, aws:SourceAccount, aws:SourceOrgID ou aws:SourceOrgPaths nas políticas de recursos sempre que uma entidade principal de serviço da AWS receber permissão para acessar um dos seus recursos. Essas chaves de condição permitem que você teste nas políticas de recursos ou nas políticas de controle de recursos se as entidades principais de serviços da AWS que acessam os recursos estão fazendo isso em nome dos recursos da AWS, das Contas da AWS ou do AWS Organizations que você espera.

- Use `aws:SourceArn` para permitir que uma entidade principal de serviço da AWS acesse os recursos em nome de um recurso específico, como uma trilha do AWS CloudTrail específica ou uma frota do AppStream.
- Use `aws:SourceAccount` para permitir que uma entidade principal de serviço da AWS acesse os recursos em nome de uma Conta da AWS específica.
- Use `aws:SourceOrgID` para permitir que a entidade principal de um serviço da AWS acesse os seus recursos em nome de determinadas AWS Organizations.
- Use `aws:SourceOrgPaths` para permitir que uma entidade principal de serviço da AWS acesse os recursos em nome de um caminho do AWS Organizations específico.

O diagrama a seguir demonstra o cenário de "confused deputy" entre serviços quando um recurso é configurado com a chave de contexto de condição global `aws:SourceAccount`, e um ator não autorizado de outra conta tenta acessar recursos da AWS aos quais não deveria ter acesso.



Usar as chaves de condição global `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` e `aws:SourceOrgPaths` em uma política ajuda a garantir que as entidades principais de serviços

estejam acessando os recursos em seu nome. Recomendamos usar essas chaves de condição sempre que o acesso a um de seus recursos for concedido a uma entidade principal de serviço da AWS.

> **Note**
>
> Algumas interações de AWS service (Serviço da AWS) têm controles adicionais para ajudar a proteger contra problemas "confused deputy" entre serviços que testam o acesso de usuários a um recurso. Por exemplo, quando uma concessão de chave do KMS é emitida para um AWS service (Serviço da AWS), o AWS KMS usa o contexto de criptografia associado ao recurso e a concessão de chave para ajudar a proteger contra problemas "confused deputy" entre serviços.
> Consulte a documentação dos serviços que você usa para obter mais informações sobre os mecanismos específicos dos serviços que podem ajudar a evitar os riscos de "confused deputy" entre serviços e se `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` e `aws:SourceOrgPaths` são compatíveis.

Proteção contra "confused deputy" entre serviços com políticas baseadas em recursos

O exemplo de política a seguir concede à entidade principal de serviço `cloudtrail.amazonaws.com` acesso ao bucket do Amazon S3, arn:aws:s3:::amzn-s3-demo-bucket1, somente quando a entidade principal de serviço está agindo em nome da Conta da AWS 111122223333.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CloudTrailAclCheck",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
```

```
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/[optionalPrefix]/Logs/
myAccountID/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

Esse exemplo de política de bucket concede à entidade principal de serviço `appstream.amazonaws.com` acesso ao script do PowerShell examplefile.psh em s3://amzn-s3-demo-bucket2 somente quando ele está agindo em nome da frota especificada do Amazon AppStream, especificando o ARN da frota com `aws:SourceArn`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "appstream.amazonaws.com"
                ]
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket2/examplefile.psh",
            "Condition": {
                "ArnEquals": {
```

```
                    "aws:SourceArn": "arn:aws:appstream:us-
east-1:111122223333:fleet/ExampleFleetName"
                }
            }
        }
    ]
}
```

Proteção contra "confused deputy" entre serviços com políticas de controle de recursos

Você pode usar políticas de controle de recursos (RCP) para aplicar controles contra "confused deputy" entre serviços aos recursos de compatíveis Serviços da AWS. As RCPs permitem que você aplique de forma centralizada controles contra "confused deputy" entre serviços em seus recursos. Você pode usar chaves de condição como `aws:SourceOrgId` e `aws:SourceOrgPaths` com RCPs anexadas ao AWS Organizations, a unidades organizacionais (OU) ou a Contas da AWS em sua organização sem adicionar instruções a políticas específicas baseadas em recursos. Para obter mais informações sobre RCPs e serviços compatíveis, consulte Políticas de Controle de Recursos (RCP) no Guia do usuário do AWS Organizations.

O exemplo a seguir de RCP nega às entidades principais de serviço da AWS o acesso aos buckets do Amazon S3 nas contas de membros quando `aws:SourceOrgID` não é igual a o-ExampleOrg. Uma permissão correspondente deve estar presente na política baseada em recursos do bucket do Amazon S3 para permitir as entidades principais do AWS service (Serviço da AWS) com um `SourceOrgID` igual a o-ExampleOrg.

Essa política aplica o controle somente nas solicitações das entidades principais de serviço (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) que tenham a chave `aws:SourceAccount` presente (`"Null": {"aws:SourceAccount": "false"}`), de modo que as integrações de serviços que não exijam o uso da chave de condição e as chamadas pelas entidades principais não sejam afetadas. Se a chave de condição `aws:SourceAccount` estiver presente no contexto da solicitação, a condição de anulação será avaliada como true, fazendo com que `aws:SourceOrgID` seja aplicada. Usamos `aws:SourceAccount` em vez de `aws:SourceOrgID` no operador de condição de anulação para que o controle ainda se seja aplicado caso a solicitação se origine de uma conta que não pertença a uma organização.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtectionForS3",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-ExampleOrg"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

Cenários comuns para perfis do IAM

Assim como ocorre com a maioria dos recursos da AWS, você geralmente tem duas maneiras de usar uma função: interativamente no console do IAM ou de forma programática com a AWS CLI, o Tools for Windows PowerShell ou a API.

- Os usuários do IAM na sua conta que estiverem usando o console do IAM podem mudar para uma função para usarem, temporariamente, as permissões da função no console. Os usuários cedem suas permissões originais e assumem as permissões atribuídas à função. Quando os usuários saem da função, suas permissões originais são restauradas.
- Uma aplicação ou um serviço oferecido pela AWS (como o Amazon EC2) pode assumir uma função ao solicitar credenciais de segurança temporárias para uma função com as quais faça

solicitações, de forma programática, para a AWS. Use uma função dessa forma para que você não precise compartilhar ou manter credenciais de segurança de longo prazo (por exemplo, criando um usuário do IAM) para cada entidade que necessite de acesso a um recurso.

> **Note**
>
> Esse guia usa as frases mudar para uma função e assumir uma função de forma intercambiável.

A maneira mais simples de usar perfis é conceder aos seus usuários do IAM permissões para mudar para os perfis que você cria dentro da sua própria conta ou em outra Conta da AWS. Eles podem mudar de funções facilmente usando o console do IAM para usar permissões que você normalmente não deseja que eles tenham e, depois, sair da função para devolver essas permissões. Isso pode ajudar a impedir o acesso acidental ou a modificação de recursos confidenciais.

Para obter mais utilizações complexas de funções, como concessão de acesso a aplicativos e serviços ou usuários externos federados, chame a API `AssumeRole`. Essa chamada de API retorna um conjunto de credenciais temporárias que o aplicativo pode usar em chamadas de API subsequentes. Ações executadas com as credenciais temporárias têm apenas as permissões concedidas pela função associada. Um aplicativo não precisa "sair" de uma função da mesma forma que um usuário no console; em vez disso, o aplicativo simplesmente é interrompido usando as credenciais temporárias e continua fazendo chamadas com as credenciais originais.

Os usuários federados fazem login usando credenciais de um provedor de identidade (IdP). A AWS fornece credenciais temporárias ao IdP confiável para serem transmitidas ao usuário e incluídas em solicitações de recursos subsequentes da AWS. Essas credenciais fornecem as permissões concedidas para a função atribuída.

Essa seção fornece uma visão geral dos seguintes cenários:

- Fornecer acesso para um usuário do IAM em uma Conta da AWS que você possui para acessar recursos em outra conta que você possui
- Fornecer acesso a workloads que não são da AWS
- Fornecer acesso a usuários do IAM em Contas da AWS de terceiros
- Fornecer acesso para serviços oferecidos pela AWS para recursos da AWS
- Fornecer acesso aos usuários autenticados externamente (federação de identidades)

Acesso a um usuário do IAM em outra Conta da AWS de sua propriedade

Você pode conceder aos usuários do IAM permissões para alternar para perfis na sua Conta da AWS ou para perfis definidos em outras Contas da AWS que você possui.

> **Note**
>
> Se deseja conceder acesso a uma conta que você não possui ou controla, veja Acesso às Contas da AWS de propriedade de terceiros mais adiante neste tópico.

Imagine que você tenha instâncias do Amazon EC2 que são críticas para sua organização. Em vez de conceder diretamente aos usuários permissão para encerrar as instâncias, você pode criar uma função com esses privilégios. Em seguida, permita que os administradores alternem para a função quando eles precisam encerrar uma instância. Isso adiciona as seguintes camadas de proteção para as instâncias:

- É necessário conceder explicitamente aos usuários permissão para assumir a função.
- Os usuários devem alternar ativamente para a função usando o Console de gerenciamento da AWS ou assumir a função usando a AWS CLI ou a API da AWS.
- Você pode adicionar a proteção de autenticação multifator (MFA) à função para que somente os usuários que fizerem login com um dispositivo MFA possam assumir a função. Para saber como configurar uma função de maneira que os usuários que assumem a função precisem primeiro ser autenticados usando a autenticação multifator (MFA), consulte Acesso seguro à API com a MFA.

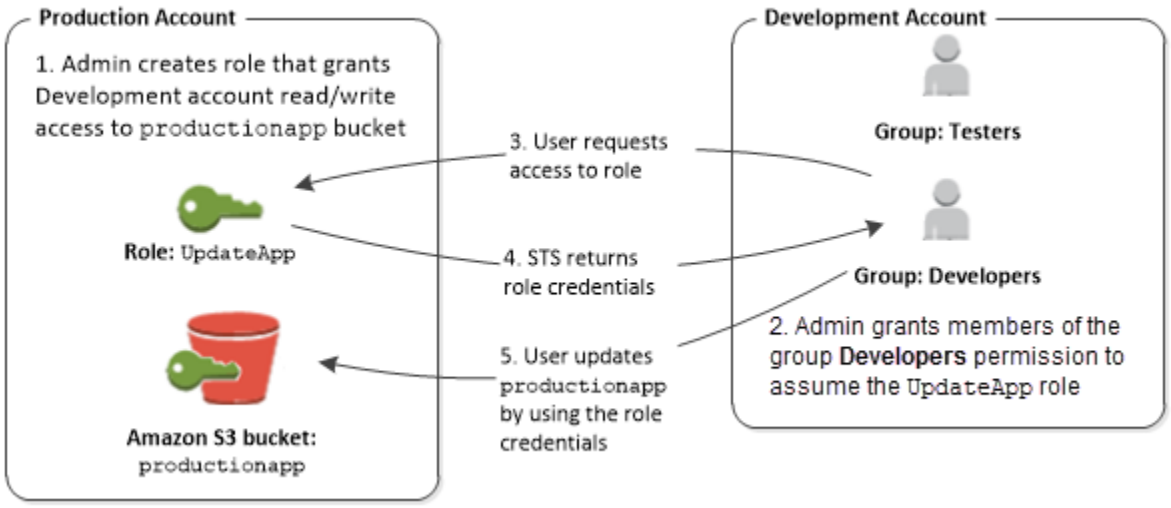
Recomendamos usar essa abordagem para aplicar o princípio do privilégio mínimo. Isso significa restringir o uso de permissões elevadas para apenas quando elas forem necessárias para tarefas específicas. Com os perfis, você pode ajudar a evitar alterações acidentais em ambientes confidenciais, especialmente se você combiná-los com uma auditoria para ajudar a certificar-se de que sejam usados apenas quando necessário.

Quando cria uma função com esse propósito, você especifica as contas pelo ID cujos usuários precisam de acesso no elemento `Principal` da política de confiança da função. Depois, você pode conceder permissões a usuários específicos nessas outras contas para alternar para a função. Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

Um usuário em uma conta pode alternar para uma função na mesma conta ou em uma diferente. Ao usar a função, o usuário pode executar somente as ações e acessar somente os recursos permitidos pela função; as permissões originais do usuário são suspensas. Quando o usuário fecha a função, as permissões originais do usuário são restauradas.

Cenário de exemplo que usa contas separadas de desenvolvimento e produção

Imagine que sua organização tenha várias Contas da AWS para isolar um ambiente de desenvolvimento de um ambiente de produção. Os usuários na conta de desenvolvimento podem precisar acessar os recursos na conta de produção. Por exemplo, você pode precisar de acesso entre contas ao promover uma atualização do ambiente de desenvolvimento para um ambiente de produção. Embora você pudesse criar identidades (e senhas) separadas para os usuários que trabalham nas duas contas, gerenciar credenciais para várias contas dificulta o gerenciamento de identidades. Na figura a seguir, todos os usuários são gerenciados na conta de desenvolvimento, mas alguns desenvolvedores exigem acesso limitado à conta de produção. A conta de desenvolvimento tem dois grupos: os testadores e os desenvolvedores, e cada grupo tem sua própria política.



1. Na conta de produção, um administrador usa o IAM para criar a função `UpdateApp` nessa conta. Na função, o administrador define uma política de confiança que especifica a conta de desenvolvimento como `Principal`, o que significa que usuários autorizados da conta de desenvolvimento podem usar a função `UpdateApp`. O administrador também define uma política de permissões para a função que especifica as permissões de leitura e gravação para o bucket do Amazon S3 denominado `productionapp`.

O administrador acaba compartilhando as informações apropriadas com qualquer pessoa que precise assumir a função. Essas informações são o número da conta e o nome da função (para usuários do console da AWS) ou o Amazon Resource Name (ARN) (para acesso à AWS CLI ou à API da AWS). O ARN da função pode ser semelhante a `arn:aws:iam::123456789012:role/UpdateApp`, em que a função se chama `UpdateApp` e foi criada na conta de número 123456789012.

> **Note**
>
> O administrador pode optar por configurar a função para que os usuários que assumirem a função precisem primeiro ser autenticados usando a autenticação multifator (MFA). Para obter mais informações, consulte Acesso seguro à API com a MFA.

2. Na conta de desenvolvimento, um administrador concede aos membros do grupo de desenvolvedores permissão para alternar para a função. Isso é feito ao conceder ao grupo de desenvolvedores a permissão para chamar a API AWS Security Token Service do AWS STS (`AssumeRole`) para a função `UpdateApp`. Qualquer usuário do IAM que pertença ao grupo de desenvolvedores na conta de desenvolvimento agora pode alternar para a função `UpdateApp` na conta de produção. Outros usuários que não estão no grupo de desenvolvedor não têm permissão para alternar para a função e, portanto, não podem acessar o bucket do S3 na conta de produção.
3. As solicitações do usuário alternam para o perfil:
   - Console da AWS: O usuário escolhe o nome da conta na barra de navegação e escolhe Mudar de função. O usuário especifica o ID da conta (ou alias) e o nome da função. Como alternativa, o usuário pode escolher um link enviado por e-mail pelo administrador. O link leva o usuário para a página Alternar função com os detalhes já preenchidos.
   - API da AWS/AWS CLI : Um usuário no grupo de desenvolvedores da conta de desenvolvimento chama a função `AssumeRole` para obter credenciais para a função `UpdateApp`. O usuário especifica o ARN da função `UpdateApp` como parte da chamada. Se um usuário no grupo de testadores faz a mesma solicitação, a solicitação falha porque os testadores não têm permissão para chamar `AssumeRole` para o ARN da função `UpdateApp`.
4. AWS STSO retorna credenciais temporárias:
   - Console da AWS: o AWS STS verifica a solicitação com a política de confiança do perfil para certificar-se de que a solicitação é de uma entidade confiável (que é a conta de desenvolvimento). Após a verificação, o AWS STS retorna as credenciais de segurança temporárias ao console da AWS.

- API/CLI: o AWS STS verifica a solicitação em relação à política de confiança do perfil para certificar-se de que a solicitação é de uma entidade confiável (que é a conta de desenvolvimento). Após a verificação, o AWS STS retorna as credenciais de segurança temporárias ao aplicativo.

5. As credenciais temporárias permitem acesso aos recursos da AWS:

- Console da AWS: o console da AWS usa as credenciais temporárias em nome do usuário para todas as ações subsequentes do console, nesse caso, para ler e gravar no bucket `productionapp`. O console não pode acessar qualquer outro recurso na conta de produção. Quando o usuário fecha a função, as permissões do usuário revertem para as permissões mantidas antes de alternar para a função.
- API/CLI: o aplicativo usa as credenciais de segurança temporárias para atualizar o bucket `productionapp`. Com as credenciais de segurança temporárias, o aplicativo só poderá ler e gravar no bucket `productionapp` e não poderá acessar qualquer outro recurso na conta de produção. O aplicativo não tem que sair da função, mas, em vez disso, interromper o uso de credenciais temporárias e usar as credenciais nas chamadas de API subsequentes.

Recursos adicionais

Para saber mais, consulte:

- Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM

Acesso para workloads que não sejam da AWS

Um perfil do IAM é um objeto no AWS Identity and Access Management (IAM) que recebe permissões. Quando você assume esse perfil usando uma identidade do IAM ou uma identidade de fora da AWS, credenciais de segurança temporárias são fornecidas para sua sessão de perfil. Você pode ter workloads em execução em seu data center ou em outra infraestrutura fora da AWS que precisem acessar seus recursos da AWS. Em vez de criar, distribuir e gerenciar chaves de acesso de longo prazo, você pode usar o AWS Identity and Access Management Roles Anywhere (IAM Roles Anywhere) para autenticar suas workloads que não são da AWS. O IAM Roles Anywhere usa certificados X.509 de sua autoridade de certificação (CA) para autenticar identidades e fornecer acesso seguro aos Serviços da AWS com as credenciais temporárias fornecidas por um perfil do IAM.

Para usar o IAM Roles Anywhere

1. Configure uma CA usando o Autoridade de certificação privada da AWS ou use uma CA de sua própria infraestrutura de PKI.
2. Depois de configurar uma CA, crie um objeto no IAM Roles Anywhere chamado âncora de confiança. Essa âncora estabelece confiança entre o IAM Roles Anywhere e sua CA para autenticação.
3. Em seguida, você pode configurar seus perfis existentes do IAM ou criar novos perfis que confiam no serviço do IAM Roles Anywhere.
4. Autentique suas workloads não AWS com o IAM Roles Anywhere usando a âncora de confiança. A AWS concede credenciais temporárias às workloads que não sejam da AWS para o perfil do IAM que tem acesso aos seus recursos da AWS.

Recursos adicionais

Os recursos a seguir podem ajudar você a saber mais sobre como fornecer acesso a workloads que não sejam da AWS.

- Para obter mais informações sobre como configurar o IAM Roles Anywhere, consulte O que é o AWS Identity and Access Management Roles Anywhere no Guia do usuário do IAM Roles Anywhere.
- Para saber como configurar a infraestrutura de chave pública (PKI) para o IAM Roles Anywhere, consulte IAM Roles Anywhere with an external certificate authority no Blog de segurança da AWS.

Acesso às Contas da AWS de propriedade de terceiros

Quando terceiros precisam de acesso a recursos da AWS da sua organização, você pode usar funções para delegar acesso a eles. Por exemplo, um terceiro pode fornecer um serviço para gerenciar seus recursos da AWS. Com funções do IAM, você pode conceder a esses terceiros acesso aos seus recursos da AWS sem compartilhar suas credenciais de segurança da AWS. Em vez disso, os terceiros podem acessar seus recursos da AWS assumindo um perfil que você cria na sua Conta da AWS. Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

Esses terceiros devem fornecer as seguintes informações para criar uma função que eles podem assumir:

- O ID da Conta da AWS do terceiro. Você especifica o ID da Conta da AWS dele como a entidade principal ao definir a política de confiança para o perfil.
- Um ID externo a ser associado de forma exclusiva ao perfil. O ID externo pode ser qualquer identificador secreto conhecido apenas por você e pelo terceiro. Por exemplo, você pode usar um ID da fatura entre você e o terceiro, mas não use algo que possa ser adivinhado, como o nome ou o número de telefone do terceiro. Você deve especificar esse ID ao definir a política de confiança para a função. O terceiro deve fornecer esse ID quando assumir a função.
- As permissões de que terceiros precisam para trabalhar com seus recursos da AWS. Você deve especificar essas permissões ao definir a política de permissões da função. Essa política define quais ações podem ser tomadas e quais recursos podem ser acessados.

Depois de criar a função, você deve fornecer o nome de recurso da Amazon (ARN) ao terceiro. Eles requerem o ARN de sua função para assumir a função.

> **Important**
>
> Quando você concede a terceiros acesso aos seus recursos da AWS, eles podem acessar qualquer recurso que você especifique na política. O uso de seus recursos é cobrado de você. Certifique-se de limitar o uso de seus recursos de forma apropriada.

ID externos para acesso de terceiros

Um ID externo permite que o usuário que está assumindo o perfil assegure as circunstâncias nas quais elas operam. Também oferece uma forma para o proprietário da conta permitir que a função seja assumida apenas em determinadas circunstâncias. A função principal do ID externo é abordar e impedir o O problema "confused deputy".

> **Important**
>
> AWSA não trata o ID externo como um segredo. Depois de criar um segredo, como um par de chaves de acesso ou uma senha na AWS, você não poderá visualizá-las novamente. O ID externo de uma função pode ser visto por qualquer pessoa com permissão para visualizar a função.

Quando devo usar um ID externo?

Use um ID externo nas seguintes situações:

- Você é proprietário de uma Conta da AWS e configurou uma perfil para um terceiro que acessa outras Contas da AWS além da sua. Você deve solicitar um ID externo ao terceiro que ele inclui quando assume a sua função. Depois, você verifica o ID externo na política de confiança da sua função. Isso garante que o terceiro externo possa assumir sua função somente quando estiver agindo em seu nome.
- Você está na posição de assumir funções em nome de clientes diferentes, como a Example Corp em nosso cenário anterior. Você deve atribuir um ID externo exclusivo para cada cliente e instruí-los a adicionar o ID externo à política de confiança de sua função. Certifique-se de sempre incluir o ID externo correto em suas solicitações para assumir funções.

  Provavelmente, você já tem um identificador exclusivo para cada um de seus clientes e esse ID exclusivo será suficiente para ser usado como ID externo. O ID externo não é um valor especial que você precisa criar, explicitamente, ou rastrear, separadamente, apenas para essa finalidade.

  Sempre especifique o ID externo em suas chamadas de API `AssumeRole`. Quando um cliente oferecer a você um ARN da função, teste se você poderá assumir a função com e sem o ID externo correto. Se você assumir a função sem o ID externo correto, não armazene o ARN da função do cliente em seu sistema. Aguarde até que o cliente tenha atualizado a política de confiança de função para exigir o ID externo correto. Dessa forma, você ajudará seus clientes a agir da forma correta, mantendo-os protegidos contra o problema confused deputy.

Exemplo de cenário usando um ID externo

Por exemplo, suponha que você decida contratar uma empresa terceirizada chamada Example Corp para monitorar sua Conta da AWS e ajudar a otimizar os custos. Para rastrear seus gastos diários, a Example Corp precisa de acesso aos seus recursos da AWS. A Example Corp também monitora muitas outras contas da AWS para outros clientes.

Não dê à Exemplo Corp acesso a um usuário do IAM e suas credenciais de longo prazo na sua conta da AWS. Em vez disso, use uma função do IAM e suas credenciais de segurança temporárias. Um perfil do IAM fornece um mecanismo que permite o acesso de terceiros aos recursos da AWS sem a necessidade de compartilhar credenciais de longo prazo (como uma chave de acesso de usuário do IAM).

Você pode usar um perfil do IAM para estabelecer uma relação de confiança entre sua Conta da AWS e a conta da Example Corp. Depois que esse relacionamento for estabelecido, um membro da conta Exemplo Corp pode chamar a API AssumeRole do AWS Security Token Service para obter credenciais de segurança temporárias. Os membros da Example Corp podem usar as credenciais para acessar recursos da AWS na sua conta.

> **Note**
>
> Para obter mais informações sobre AssumeRole e outras APIs da AWS que você pode chamar para obter credenciais de segurança temporárias, consulte Compare credenciais do AWS STS.

Veja aqui um detalhamento desse cenário:

1. Contrate a Example Corp, para que eles criem um identificador de clientes exclusivo para você. Eles fornecerão o ID de cliente exclusivo e o número da Conta da AWS deles. Essas informações são necessárias para criar uma função do IAM na próxima etapa.

   > **Note**
   >
   > A Example Corp pode usar qualquer valor de string que desejar para o ExternalId, desde que seja exclusivo para cada cliente. Pode ser o número da conta de um cliente ou até mesmo uma string aleatória de caracteres, desde que dois clientes não tenham o mesmo valor. Isso não deve ser um "segredo". A Example Corp deve fornecer o valor do ExternalId para cada cliente. O essencial é que ele deve ser gerado pela Example Corp e não pelos clientes dela para garantir que cada ID externo seja exclusivo.

2. Faça login na AWS e crie uma função do IAM que dê à Exemplo Corp acesso aos seus recursos. Como qualquer função do IAM, a função tem duas políticas, uma política de permissão e uma política de confiança. A política de confiança da função especifica quem pode assumir a função. Em nosso cenário de exemplo, a política especifica o número de Conta da AWS da Example Corp como `Principal`. Isso permite que as identidades dessa conta assumam a função. Além disso, você adiciona um elemento `Condition` à política de confiança. Esse elemento `Condition` testa a chave de contexto `ExternalId` para garantir que ela corresponda ao ID do cliente exclusivo da Example Corp. Por exemplo:

   ```
   "Principal": {"AWS": "Example Corp's Conta da AWS ID"},
   ```

```
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. A política de permissões da função especifica o que a função permite que alguém faça. Por exemplo, você pode especificar que a função permita que alguém gerencie apenas seus recursos do Amazon EC2 e Amazon RDS, mas não seus usuários ou grupos do IAM. Em nosso cenário de exemplo, use a política de permissões para dar acesso somente leitura à Example Corp a todos os recursos na sua conta.
4. Depois de criar a função, forneça o nome de recurso da Amazon (ARN) da função à Example Corp.
5. Quando a Exemplo Corp precisar acessar seus recursos da AWS, alguém da empresa chamará a API `sts:AssumeRole` da AWS. A chamada inclui o ARN da função a ser assumida e o parâmetro ExternalId que corresponde ao ID do cliente.

Se a solicitação vier de alguém que estiver usando a Conta da AWS da Example Corp, e se o ARN do perfil e o ID externo estiverem corretos, a solicitação será bem-sucedida. Nesse caso, ela fornecerá credenciais de segurança temporárias que a Example Corp poderá usar para acessar os recursos da AWS permitidos pela sua função.

Em outras palavras, quando uma política de função incluir um ID externo, qualquer pessoa que desejar assumir a função deverá ser especificada como um principal na função e deverá incluir o ID externo correto.

Pontos-chave para IDs externos

- Em um ambiente de vários locatários onde você oferece suporte a vários clientes com AWS contas diferentes, recomendamos usar um ID externo por Conta da AWS. Esse ID deve ser uma string aleatória gerada por um terceiro.
- Para exigir que o terceiro forneça um ID externo ao assumir uma função, atualize a política de confiança da função com o ID externo de sua escolha.
- Para fornecer um ID externo quando você assumir uma função, use a AWS CLI ou a API da AWS para assumir essa função. Para obter mais informações, consulte a operação da API STS AssumeRole ou a operação da CLI assume-role STS.
- O valor `ExternalId` deve ter no mínimo 2 e no máximo 1.224 caracteres. O valor deve ser alfanumérico sem espaço em branco. Ele também pode incluir os seguintes símbolos: mais (+), igual (=), vírgula (,), ponto (.), arroba (@), dois pontos (:), barra (/) e hífen (-).

Recursos adicionais

Os seguintes recursos podem ajudar você a saber mais sobre como fornecer acesso a Contas da AWS de terceiros.

- Para saber mais sobre como permitir que outras pessoas realizem ações na sua Conta da AWS, consulte Criar um perfil usando políticas de confiança personalizadas.
- Para saber mais sobre como conceder permissão para alternar para um perfil, consulte Conceder permissões a um usuário para alternar perfis
- Para saber mais sobre como criar e fornecer credenciais de segurança temporárias aos usuários confiáveis, consulte Permissões de credenciais de segurança temporárias.

Acesso a um serviço da AWS

Muitos serviços da AWS exigem que você use funções para controlar o que esse serviço pode acessar. A função de serviço é uma função que um serviço assume para realizar ações em seu nome. Quando uma função atende a uma finalidade especializada para um serviço, ela pode ser categorizada como uma função vinculada ao serviço. Consulte a documentação da AWS de cada serviço para ver se ele usa funções e saber como atribuir uma função ao serviço a ser usado.

Para obter detalhes sobre a criação de uma função para delegar acesso a um serviço oferecido pela AWS, consulte Criar um perfil para delegar permissões a um serviço da AWS.

Acesso aos usuários autenticados externamente (federação de identidades)

Seus usuários podem já ter identidades fora da AWS, por exemplo, no diretório corporativo. Se esses usuários precisarem trabalhar com recursos da AWS (ou trabalhar com aplicações que acessem esses recursos), os usuários também precisarão de credenciais de segurança da AWS. Você pode usar uma função do IAM para especificar permissões para usuários cuja identidade é federada de sua organização ou de um provedor de identidade (IdP) de terceiros.

> **Note**
>
> Como prática recomendada de segurança, recomendamos que você gerencie o acesso do usuário no Centro de identidade do IAM com federação de identidades em vez de criar usuários do IAM. Para obter informações sobre situações específicas em que um usuário do IAM é necessário, consulte Quando criar um usuário do IAM (em vez de um perfil).

Os conjuntos de permissões do IAM Identity Center provisionam modelos de perfil do IAM que o IAM Identity Center gerencia. O gerenciador de acesso à conta — um atributo do IAM — permite atribuir os próprios perfis do IAM personalizados aos usuários e grupos do IAM Identity Center, habilitando recursos como políticas de confiança personalizadas, tags de perfil e caminhos de perfil configuráveis. Você pode usá-lo junto com os conjuntos de permissões ou sozinho.

Federação de usuários de um aplicativo móvel ou baseado na Web com o Amazon Cognito

Se você criar um aplicativo móvel ou baseado na Web que acesse recursos da AWS, ele precisará de credenciais de segurança para fazer solicitações programáticas à AWS. Para a maioria dos cenários de aplicações móveis, recomendamos usar o Amazon Cognito. Você pode usar esse serviço com o AWS Mobile SDK para iOS e o AWS Mobile SDK para Android e Fire OS a fim de criar identidades exclusivas para os usuários e autenticá-las para assegurar o acesso aos seus recursos da AWS. O Amazon Cognito oferece suporte aos mesmos provedores de identidade listados na próxima seção e também oferece suporte a identidades autenticadas pelo desenvolvedor e acesso não autenticado (de convidado). O Amazon Cognito também fornece operações de API para sincronização de dados de usuário para que eles sejam preservados à medida que passarem de um dispositivo para outro. Para obter mais informações, consulte Amazon Cognito para aplicativos móveis.

Federação de usuários com provedores de serviços de identidade pública ou OpenID Connect

Sempre que possível, use o Amazon Cognito para cenários de aplicativos móveis e baseados na Web. O Amazon Cognito faz a maior parte do trabalho em segundo plano com os serviços do provedor de identidade pública para você. Ele funciona com os mesmos serviços de terceiros e também dá suporte a logins anônimos. No entanto, para cenários mais avançados, você pode trabalhar diretamente com um serviço de terceiros, como o Login with Amazon, Facebook, Google ou qualquer provedor compatível com o OpenID Connect (OIDC). Para obter mais informações sobre como usar a federação OIDC com um desses serviços, consulte Federação OIDC.

Federação de usuários com SAML 2.0

Se sua organização já usa um pacote de software de provedor de identidade que ofereça suporte a SAML 2.0 (Security Assertion Markup Language 2.0), você poderá criar confiança entre a organização como provedor de identidade (IdP) e a AWS como o provedor de serviços. Em seguida, você pode usar SAML para fornecer aos usuários logon único (SSO) federado ao Console de gerenciamento da AWS ou acesso federado para chamar operações de API da AWS. Por exemplo, se sua empresa usa o Microsoft Active Directory e o Active Directory Federation Services, você

poderá realizar a federação usando o SAML 2.0. Para obter mais informações sobre como federar usuários com SAML 2.0, consulte Federação SAML 2.0.

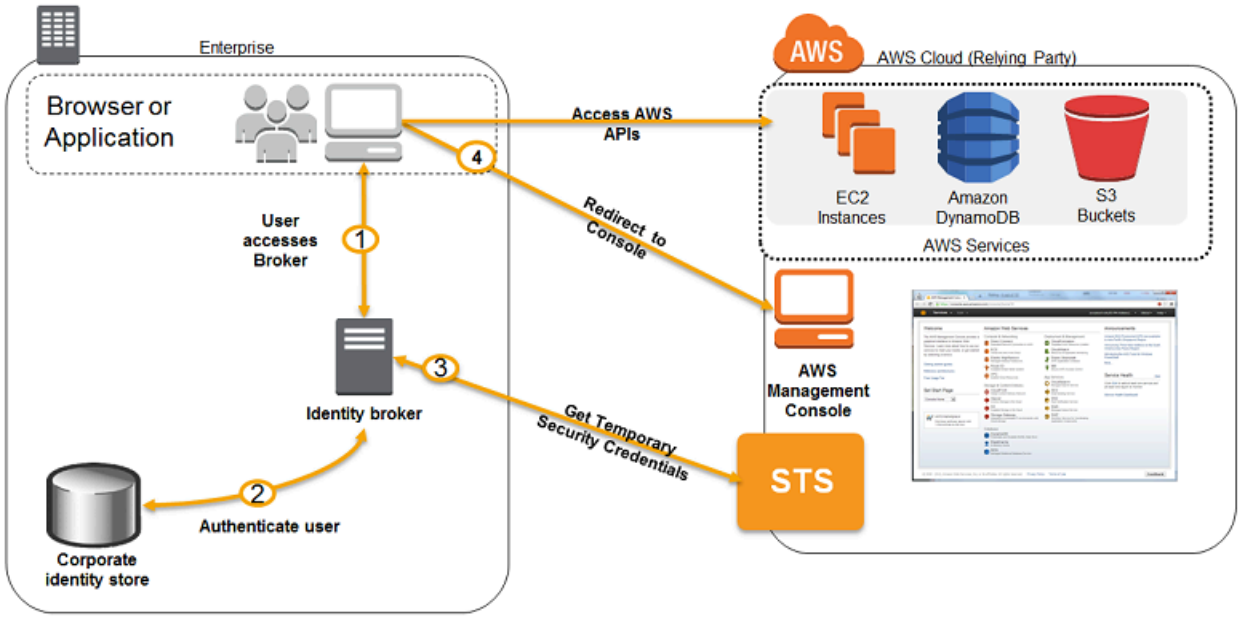
Federação de usuários criando um aplicativo identity broker personalizado

Se o armazenamento de identidades não for compatível com o SAML 2.0, você poderá criar um aplicativo identity broker personalizado para executar uma função semelhante. O aplicativo agente autentica os usuários, solicita credenciais temporárias para usuários da AWS e, em seguida, as fornece ao usuário para acessar os recursos da AWS.

Por exemplo, a Corp. de Exemplo tem muitos funcionários que precisam executar aplicativos internos que acessam os recursos da AWS da empresa. Os funcionários já têm identidades no sistema de autenticação e identidade da empresa, e a Corp. de Exemplo não deseja criar um usuário do IAM separado para cada funcionário da empresa.

Bob é um desenvolvedor na Corp. de Exemplo. Para permitir que os aplicativos internos da Corp. de Exemplo acessem os recursos da AWS da empresa, Bob desenvolve um aplicativo identity broker personalizado. O aplicativo confirma que os funcionários existentes estão conectados ao sistema de autenticação e identidade da Corp. de Exemplo, que pode usar LDAP, Active Directory ou outro sistema. O aplicativo identity broker, em seguida, obtém credenciais de segurança temporárias para os funcionários. Esse cenário é semelhante ao anterior (um aplicativo móvel que usa um sistema de autenticação personalizada), exceto pelo fato de que as aplicações que precisam de acesso a todos os recursos da AWS são executados dentro da rede corporativa, e a empresa tem um sistema de autenticação.

Para obter credenciais de segurança temporárias, o aplicativo identity broker chama `AssumeRole` ou `GetFederationToken` para obter credenciais de segurança temporárias, dependendo de como Bob precisa gerenciar as políticas para os usuários e quando as credenciais temporárias devem expirar. (Para obter mais informações sobre as diferenças entre essas operações de API, veja Credenciais de segurança temporárias no IAM e Permissões de credenciais de segurança temporárias.) A chamada retorna as credenciais de segurança temporárias que consistem em um ID da chave de acesso da AWS, uma chave de acesso secreta e um token de sessão. O aplicativo identity broker torna essas credenciais temporárias de segurança disponíveis para os aplicativos internos da empresa. Em seguida, o aplicativo pode usar as credenciais temporárias para fazer chamadas para a AWS diretamente. O aplicativo armazena as credenciais em cache até elas expirarem e, em seguida, solicita um novo conjunto de credenciais temporárias. A figura a seguir ilustra esse cenário.



Esse cenário tem os seguintes atributos:

- A aplicação do intermediador de identidades tem permissões para acessar a API do serviço de token (STS) do IAM para criar credenciais de segurança temporárias.
- O aplicativo identity broker é capaz de verificar se os funcionários estão autenticados dentro do sistema de autenticação existente.
- Os usuários podem obter um URL temporário que forneça a elrd acesso ao Console de Gerenciamento da AWS (chamada de autenticação única).

Para obter informações sobre a criação de credenciais de segurança temporárias, consulte Compare credenciais do AWS STS. Para obter mais informações sobre como as entidades principais federadas SAML obtêm acesso ao Console de Gerenciamento da AWS, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS.

Criar um perfil do IAM

Para criar uma função você pode usar o Console de gerenciamento da AWS, a AWS CLI, o Tools for Windows PowerShell ou a API do IAM.

Se você usar o Console de gerenciamento da AWS, um assistente orienta você durante as etapas para a criação de uma função. O assistente tem etapas ligeiramente diferentes dependendo se você esta criando um perfil para um serviço da AWS, para uma Conta da AWS ou para uma entidade principal federada SAML ou OIDC.

Perfis para usuários do IAM

Crie esse perfil para delegar permissões na sua Conta da AWS ou para perfis definidos em outras Contas da AWS que você possui. Um usuário em uma conta pode alternar para uma função na mesma conta ou em uma diferente. Ao usar a função, o usuário pode executar somente as ações e acessar somente os recursos permitidos pela função; as permissões originais do usuário são suspensas. Quando o usuário fecha a função, as permissões originais do usuário são restauradas.

Para obter mais informações, consulte Criar um perfil para conceder permissões a um usuário do IAM.

Para obter mais informações sobre a criação de perfis para acesso entre contas, consulte Criar um perfil usando políticas de confiança personalizadas.

Perfis para serviços da AWS

Crie esse perfil para delegar permissões a um serviço que pode realizar ações em seu nome. Um perfil de serviço que você passa para um serviço deve ter uma política do IAM com as permissões que possibilitem que o serviço realize ações associadas a esse serviço. Diferentes permissões são necessárias para cada um dos serviços da AWS.

Para obter mais informações sobre como criar perfis de serviço, consulte Criar um perfil para delegar permissões a um serviço da AWS.

Para obter mais informações sobre como criar perfis vinculados a serviço, consulte Criar um perfil vinculado ao serviço.

Perfis para federação de identidades

Crie esse perfil para delegar permissões a usuários que já têm identidades externas à AWS. Quando você usa um provedor de identidade do , não precisa criar código de login personalizado nem gerenciar suas próprias identidades de usuários. Os usuários externos fazem login por meio do IdP, e você pode conceder a essas identidades externas permissões para usar os recursos da AWS na sua conta. Os provedores de identidade ajudam a manter sua conta da AWS segura, pois você não precisa distribuir nem incorporar credenciais de segurança de longo prazo, como chaves de acesso, em sua aplicação.

Para obter mais informações, consulte Criar um perfil para um provedor de identidade de terceiros.

Criar um perfil para conceder permissões a um usuário do IAM

É possível usar perfis do IAM para delegar acesso aos seus recursos da AWS. Com funções do IAM, você pode estabelecer relações de confiança entre sua conta de confiança e outras contas confiáveis da AWS. A conta de confiança tem o recurso a ser acessado e a conta confiável contém os usuários que precisam de acesso ao recurso. No entanto, é possível que outra conta tenha um recurso em sua conta. Por exemplo, a conta de confiança pode permitir que a conta confiável crie novos recursos, como a criação de novos objetos em um bucket do Amazon S3. Nesse caso, a conta que cria o recurso possui o recurso e controla quem pode acessar esse recurso.

Depois de criar a relação de confiança, um usuário do IAM ou um aplicativo da conta confiável pode usar a operação `AssumeRole` da API do AWS Security Token Service (AWS STS). Essa operação fornece credenciais de segurança temporárias que permitem acesso aos recursos da AWS em sua conta.

As contas podem ser controladas por você ou a conta com os usuários pode ser controlada por terceiros. Se a outra conta com os usuários for uma Conta da AWS controlada por você, use o atributo `externalId`. O ID externo pode ser qualquer palavra ou número combinado entre você e o administrador da conta de terceiros. Esta opção adiciona automaticamente uma condição à política de confiança que permite ao usuário assumir a função somente se a solicitação incluir o correto `sts:ExternalID`. Para obter mais informações, consulte Acesso às Contas da AWS de propriedade de terceiros.

Para obter informações sobre como usar funções para delegar permissões, consulte Termos e conceitos das funções. Para obter mais informações sobre o uso de uma função de serviço para permitir que os serviços acessem recursos na sua conta, consulte Criar um perfil para delegar permissões a um serviço da AWS.

Criação de uma função do IAM (console)

Você pode usar o Console de gerenciamento da AWS para criar uma função que um usuário do IAM pode assumir. Por exemplo, suponha que sua organização tem várias Contas da AWS para isolar um ambiente de desenvolvimento de um ambiente de produção. Para obter informações de alto nível sobre a criação de uma função que permita que os usuários na conta de desenvolvimento acessem recursos na conta de produção, consulte Cenário de exemplo que usa contas separadas de desenvolvimento e produção.

> **Permissões mínimas**
>
> Para executar as etapas a seguir, é necessário ter as seguintes permissões do IAM:

- `access-analyzer:ValidatePolicy`
- `iam:AttachRolePolicy`
- `iam:CreatePolicy`
- `iam:CreateRole`
- `iam:GetAccountSummary`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:ListAccountAliases`
- `iam:ListAttachedRolePolicies`
- `iam:ListOpenIDConnectProviders`
- `iam:ListPolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`
- `iam:ListRoleTags`
- `iam:ListSAMLProviders`

Console

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Roles (Perfis) e, em seguida, clique em Create role (Criar perfil).
3. Escolha o tipo de perfil do Conta da AWS.
4. Para criar uma função para sua conta, escolha This account (Esta conta). Para criar um perfil para outra conta, escolha Outra Conta da AWS e insira o ID da conta para o qual você deseja conceder acesso a seus recursos.

   O administrador da conta especificada pode conceder permissão para assumir essa função a qualquer usuário do IAM dessa conta. Para fazer isso, o administrador anexa uma política ao usuário ou grupo que concede permissão para a ação `sts:AssumeRole`. Essa política deve especificar o ARN da função como `Resource`.

5. Se estiver concedendo permissões aos usuários de uma conta que você não controla e os usuários assumirem essa função de maneira programática, selecione Require external ID (Exigir ID externo). O ID externo pode ser qualquer palavra ou número combinado entre você e o administrador da conta de terceiros. Esta opção adiciona automaticamente uma condição à política de confiança que permite ao usuário assumir a função somente se a solicitação incluir o correto `sts:ExternalID`. Para obter mais informações, consulte Acesso às Contas da AWS de propriedade de terceiros.

   > **Important**
   >
   > A escolha dessa opção restringe o acesso à função apenas por meio da AWS CLI, do Tools for Windows PowerShell ou da API da AWS. A razão disso é que você não pode usar o console da AWS para alternar para uma função que tenha uma condição `externalId` na sua política de confiança. No entanto, você pode criar esse tipo de acesso de modo programático ao escrever um script ou um aplicativo usando o SDK relevante. Para obter mais informações e um script de exemplo, consulte How to Enable Cross-Account Access to the Console de gerenciamento da AWS no AWS Security Blog.

6. Se você desejar restringir a função a usuários que façam login usando um dispositivo de multi-factor authentication (MFA), selecione a opção Exigir MFA. Isso adiciona uma condição à política de confiança da função que verifica a existência de um login MFA. Um usuário que desejar assumir a função deverá fazer login com uma senha de uso único temporária a partir de um dispositivo MFA configurado. Os usuários sem a autenticação por MFA não podem assumir a função. Para obter mais informações sobre MFA, consulte AWSCódigo da autenticação multifator no IAM da
7. Escolha Próximo.
8. O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta. Selecione a política a ser usada para a política de permissões ou escolha Create policy (Criar política) para abrir uma nova guia no navegador e criar uma nova política a partir do zero. Para obter mais informações, consulte Criação de políticas do IAM. Depois de criar a política, feche essa guia e retorne à guia original. Marque a caixa de seleção ao lado das políticas de permissões que você deseja que qualquer pessoa que assuma a função tenha. Se preferir, você pode optar por não selecionar nenhuma política neste momento e anexar as políticas à função mais tarde. Por padrão, uma função não tem nenhuma permissões.
9. (Opcional) Defina um limite de permissões. Este é um recurso avançado.

Abra a seção Set permissions boundary (Definir limite de permissões) e escolha Use a permissions boundary to control the maximum role permissions (Usar um limite de permissões para controlar o número máximo de permissões de funções). Selecione a política a ser usada para o limite de permissões.

10. Escolha Próximo.
11. Em Role name (Nome da função), digite um nome para sua função. Os nomes de função devem ser exclusivos em sua Conta da AWS. Ao ser usado em uma política ou como parte de um ARN, o nome de perfil diferencia maiúsculas de minúsculas. Quando exibida para os clientes no console, por exemplo, como durante o processo de login, o nome de função não diferencia maiúsculas de minúsculas. Como várias entidades podem fazer referência à função, não é possível editar o nome da função depois de criada.
12. (Opcional) Em Descrição da função, insira uma descrição para a nova função.
13. Escolha Edit (Editar) nas seções Etapa 1: selecionar entidades confiáveis ou na Etapa 2: adicionar permissões para editar os casos de uso e as permissões para a função. Você será levado de volta às páginas anteriores para fazer as edições.
14. (Opcional) Adicione metadados à função anexando tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
15. Revise o perfil e escolha Criar perfil.

> **Important**
>
> Lembre-se de que esta é apenas a metade da configuração necessária. Você também deve conceder aos usuários individuais na conta confiável permissões para alternar para a função no console ou assumir a função de forma programática. Para obter mais informações sobre essa etapa, consulte Conceder permissões a um usuário para alternar perfis.

Criação de uma função do IAM (AWS CLI)

A criação de uma função a partir da AWS CLI envolve várias etapas. Quando o console é usado para criar uma função, muitas das etapas são concluídas por você, mas com a AWS CLI, é necessário executar explicitamente cada etapa. Você deve criar a função e atribuir uma política de permissões à função. Opcionalmente, você também pode definir o limite de permissões para sua função.

Para criar uma função para acesso entre contas (AWS CLI)

1. Criar uma função: aws iam create-role
2. Anexar uma política de permissões gerenciada à função: aws iam attach-role-policy

   ou

   Criar uma política de permissões em linha para a função: aws iam put-role-policy
3. (Opcional) Adicione atributos personalizados à função anexando tags: aws iam tag-role

   Para obter mais informações, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).
4. (Opcional) Definir o limite de permissões para a função: aws iam put-role-permissions-boundary

   Um limite de permissões controla o número máximo de permissões que uma função pode ter. Os limites de permissões são um recurso avançado da AWS.

O exemplo a seguir mostra as duas primeiras e as etapas mais comuns para a criação de uma função entre contas em um ambiente simples. Este exemplo permite que qualquer usuário na conta `123456789012` assuma a função e visualize o bucket `example_bucket` do Amazon S3. Este exemplo também supõe que você está em um computador cliente com o Windows e já configurou a interface de linha de comando com as credenciais de sua conta e região. Para obter mais informações, consulte Configurar a interface de linha de comando da AWS.

Neste exemplo, inclua a seguinte política de confiança no primeiro comando ao criar a função. A política de confiança permite que os usuários na conta `123456789012` assumam a função usando a operação `AssumeRole`, mas apenas se o usuário fornecer autenticação MFA usando os parâmetros `SerialNumber` e `TokenCode`. Para obter mais informações sobre MFA, consulte AWSCódigo da autenticação multifator no IAM da.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
```

```
            "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
        }
    ]
}
```

> **Important**
>
> Se seu elemento `Principal` contiver o ARN de uma função ou um usuário específico do IAM, esse ARN será transformado em um ID de entidade de segurança exclusivo quando a política for salva. Isso ajuda a reduzir o risco de alguém elevar suas permissões ao remover e recriar a função ou usuário. Normalmente, você não vê esse ID no console, porque há também uma transformação reversa de volta para o ARN quando a política de confiança é exibida. No entanto, se você excluir a função ou o usuário, o ID da entidade principal aparecerá no console porque a AWS não pode mais mapeá-lo de volta para um ARN. Portanto, se você excluir e recriar um usuário ou função referenciado no elemento `Principal` de uma política de confiança, você deverá editar a função para substituir o ARN.

Quando usa o segundo comando, você deve anexar uma política gerenciada existente à função. A política de permissões a seguir permite que qualquer pessoa que assuma a função execute apenas a ação `ListBucket` no bucket `example_bucket` do Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
  ]
}
```

Para criar essa `Test-UserAccess-Role` função, você deve primeiro salvar a política de confiança anterior com o nome `trustpolicyforacct123456789012.json` na pasta `policies` em sua

`C:` unidade local. Em seguida, salve a política de permissões anterior como uma política gerenciada pelo cliente na sua Conta da AWS com o nome `PolicyForRole`. Você pode usar os comandos a seguir para criar a função e anexar a política gerenciada.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> Lembre-se de que esta é apenas a metade da configuração necessária. Você também deve conceder aos usuários individuais na conta confiável permissões para alternar para a função. Para obter mais informações sobre essa etapa, consulte Conceder permissões a um usuário para alternar perfis.

Depois de criar a função e conceder permissões a ela para executar tarefas da AWS ou acessar recursos da AWS, qualquer usuário da conta `123456789012` poderá assumir a função. Para obter mais informações, consulte Alterar para uma perfil do IAM (AWS CLI).

Criação de uma função do IAM (API da AWS)

A criação de uma função na API da AWS envolve várias etapas. Quando o console é usado para criar uma função, muitas das etapas são concluídas por você, mas com a API, é necessário executar explicitamente cada etapa. Você deve criar a função e atribuir uma política de permissões à função. Opcionalmente, você também pode definir o limite de permissões para sua função.

Para criar uma função em código (API da AWS)

1. Criar uma função: CreateRole

   Para a política de confiança da função, você pode especificar um local de arquivo.
2. Anexar uma política de permissões gerenciada à função: AttachRolePolicy

ou

Criar uma política de permissões embutida para a função: PutRolePolicy

> **Important**
>
> Lembre-se de que esta é apenas a metade da configuração necessária. Você também deve conceder aos usuários individuais na conta confiável permissões para alternar para a função. Para obter mais informações sobre essa etapa, consulte Conceder permissões a um usuário para alternar perfis.

3. (Opcional) Adicione atributos personalizados ao perfil anexando tags: TagRole

   Para obter mais informações, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).
4. (Opcional) Definir o limite de permissões para a função: PutRolePermissionsBoundary

   Um limite de permissões controla o número máximo de permissões que uma função pode ter. Os limites de permissões são um recurso avançado da AWS.

Depois de criar a função e conceder permissões a ela para executar tarefas da AWS ou acessar recursos da AWS, você deve conceder permissões aos usuários da conta para permitir que eles assumam a função. Para obter mais informações sobre a assunção de uma função, consulte Alternar para um perfil do IAM (API da AWS).

Criação de uma função do IAM (AWS CloudFormation)

Para obter informações sobre como criar uma função do IAM no AWS CloudFormation, consulte a referência de recursos e propriedades e exemplos no Guia do usuário do AWS CloudFormation.

Para obter mais informações sobre os modelos do IAM no AWS CloudFormation, consulte Trechos de modelo do AWS Identity and Access Management no Guia do usuário do AWS CloudFormation.

Criar um perfil para delegar permissões a um serviço da AWS

Muitos serviços da AWS exigem que você use funções para permitir que o serviço acesse recursos em outros serviços em seu nome. A função de serviço é uma função que um serviço assume para realizar ações em seu nome. Quando um perfil atende a uma finalidade especializada para um serviço, ela é categorizada como um perfil vinculado ao serviço. Para ver quais serviços oferecem

suporte ao uso de funções vinculadas a serviços, ou se um serviço oferece suporte a qualquer forma de credenciais temporárias, consulte AWSServiços da que funcionam com o IAM. Para saber como determinado serviço usa funções, escolha o nome do serviço na tabela para visualizar a documentação dele.

Ao definir a permissão `PassRole`, é necessário garantir que um usuário não passe um perfil em que o perfil tenha mais permissões do que você deseja que o usuário tenha. Por exemplo, Alice pode não ter permissão para realizar nenhuma ação do Amazon S3. Se Alice pudesse passar um perfil para um serviço que permite ações do Amazon S3, o serviço poderia realizar ações do Amazon S3 em nome de Alice ao executar o trabalho.

Para obter informações sobre como as funções ajudam você a delegar permissões, consulte Termos e conceitos das funções.

Permissões de função de serviço

Configure permissões para que uma entidade do IAM (usuário ou função) crie ou edite uma função de serviço.

> **Note**
>
> O ARN de uma função vinculado ao serviço inclui uma entidade principal do serviço que é indicada nas políticas a seguir como *`SERVICE-NAME`*`.amazonaws.com`. Não tente adivinhar a entidade principal do serviço, pois ela faz distinção entre maiúsculas e minúsculas, e o formato pode variar entre os serviços da AWS. Para visualizar a entidade principal do serviço, consulte a documentação do perfil vinculado ao serviço.

Para permitir que uma entidade do IAM crie uma função de serviço específica

Adicione a seguinte política à entidade do IAM que precisa criar a função de serviço. Essa política permite que você crie uma função de serviço para o serviço especificado e com um nome específico. Em seguida, você poderá anexar políticas em linha ou gerenciadas a essa função.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

Para permitir que uma entidade do IAM crie qualquer função de serviço

AWSA recomenda permitir apenas que usuários administrativos criem qualquer perfil de serviço. Uma pessoa com permissões para criar uma função e anexar qualquer política pode aumentar as próprias permissões. Em vez disso, crie uma política que permita criar apenas as funções necessárias ou peça para um administrador criar a função de serviço no nome dessas pessoas.

Para anexar uma política que permita que um administrador acesse toda a sua Conta da AWS, use a política gerenciada AdministratorAccess da AWS.

Para permitir que uma entidade do IAM edite uma função de serviço

Adicione a seguinte política à entidade do IAM que precisa editar a função de serviço.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
```

```
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

Para permitir que uma entidade do IAM exclua uma função de serviço específica

Adicione a seguinte instrução à política de permissões à entidade do IAM que precisa excluir a função de serviço especificada.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

Para permitir que uma entidade do IAM exclua qualquer função de serviço

AWSA recomenda permitir apenas que usuários administrativos excluam qualquer perfil de serviço. Em vez disso, crie uma política que permita excluir apenas as funções necessárias ou peça para um administrador excluir a função de serviço no nome dessas pessoas.

Para anexar uma política que permita que um administrador acesse toda a sua Conta da AWS, use a política gerenciada AdministratorAccess da AWS.

Criar uma função para um serviço da AWS (console)

É possível usar o Console de gerenciamento da AWS para criar um perfil para um serviço. Como alguns serviços oferecem suporte a mais de uma função de serviço, consulte a Documentação

da AWS para o seu serviço para ver qual caso de uso escolher. É possível saber como atribuir as políticas de confiança e as permissões necessárias ao perfil para que o serviço possa assumir o perfil em seu nome. As etapas que podem se usadas para controlar as permissões para a sua função podem variar, dependendo de como o serviço define os casos de uso e de se você cria ou não uma função vinculada ao serviço.

Console

Para criar uma função para um AWS service (Serviço da AWS) (console do IAM)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de entidade confiável, escolha AWS service (Serviço da AWS).
4. Para Serviço ou caso de uso, escolha um serviço e, em seguida, escolha o caso de uso. Casos de uso são definidos pelo serviço para incluir a política de confiança exigida pelo serviço.
5. Escolha Próximo.
6. As opções para Políticas de permissões dependem do caso de uso selecionado.
   - Se o serviço definir as permissões para o perfil, não será possível selecionar políticas de permissões.
   - Selecione em um conjunto limitado de políticas de permissões.
   - Selecione entre todas as políticas de permissões.
   - Não selecione nenhuma política de permissão; crie políticas após a criação do perfil e, em seguida, anexe as políticas ao perfil.
7. (Opcional) Defina um limite de permissões. Esse é um atributo avançado que está disponível para perfis de serviço, mas não para perfis vinculados ao serviço.
   a. Abra a seção Definir limite de permissões e escolha Usar um limite de permissões para controlar o número máximo de permissões do perfil.

      O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta.
   b. Selecione a política a ser usada para o limite de permissões.
8. Escolha Próximo.
9. Para Nome do perfil, as opções dependem do serviço:

- Se o serviço definir o nome do perfil, não será possível editar esse nome.
- Se o serviço definir um prefixo para o nome do perfil, você poderá inserir um sufixo opcional.
- Se o serviço definir o nome do perfil, você poderá atribuir um nome ao perfil.

> **Important**
>
> Quando nomear um perfil, observe o seguinte:
>
> - Os nomes do perfil devem ser exclusivos em sua Conta da AWS e não podem ser diferenciados caso a caso.
>
> Por exemplo, não crie dois perfis denominados **`PRODROLE`** e **`prodrole`**. Quando usado em uma política ou como parte de um ARN, o nome de perfil diferencia maiúsculas de minúsculas. No entanto, quando exibido para os clientes no console, como durante o processo de login, o nome de perfil diferencia maiúsculas de minúsculas.
> - Não é possível editar o nome do perfil depois de criá-lo porque outras entidades podem referenciar o perfil.

10. (Opcional) Em Descrição, insira uma descrição para o perfil.
11. (Opcional) Para editar os casos de uso e as permissões do perfil, escolha Editar nas seções Etapa 1: selecionar entidades confiáveis ou Etapa 2: adicionar permissões.
12. (Opcional) Para ajudar a identificar, organizar ou pesquisar o perfil, adicione tags como pares de chave-valor. Para obter mais informações sobre o uso de tags no IAM, consulte Tags para recursos do AWS Identity and Access Management no Guia do usuário do IAM.
13. Reveja a função e escolha Criar função.

Criar uma função para um serviço (AWS CLI)

A criação de uma função a partir da AWS CLI envolve várias etapas. Quando o console é usado para criar uma função, muitas das etapas são concluídas por você, mas com a AWS CLI, é necessário executar explicitamente cada etapa. Você deve criar a função e atribuir uma política de permissões à função. Se o serviço com o qual você está trabalhando for o Amazon EC2, você também deverá criar um perfil de instância e adicionar a função a ele. Opcionalmente, você também pode definir o limite de permissões para sua função.

Para criar uma função para um serviço da AWS na AWS CLI

1. O seguinte comando `create-role` cria uma função chamada Test-Role e anexa uma política de confiança a ela:

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. Anexe uma política de permissões gerenciadas à função: aws iam attach-role-policy.

   Por exemplo, o seguinte comando `attach-role-policy` anexa a política gerenciada pela AWS chamada `ReadOnlyAccess` à função do IAM chamada `ReadOnlyRole`:

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   ReadOnlyAccess --role-name ReadOnlyRole
   ```

   ou

   Criar uma política de permissões em linha para a função: aws iam put-role-policy

   Para adicionar uma política de permissões em linha, consulte o seguinte exemplo:

   ```
   aws iam put-role-policy --role-name Test-Role --policy-name
   ExamplePolicy --policy-document file://AdminPolicy.json
   ```

3. (Opcional) Adicione atributos personalizados à função anexando tags: aws iam tag-role

   Para obter mais informações, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).

4. (Opcional) Definir o limite de permissões para a função: aws iam put-role-permissions-boundary

   Um limite de permissões controla o número máximo de permissões que uma função pode ter. Os limites de permissões são um recurso avançado da AWS.

Se você for usar a função com o Amazon EC2 ou outro produto da AWS que use o Amazon EC2, armazene a função em um perfil de instância. Um perfil da instância é um contêiner para uma função que pode ser anexado a uma instância do Amazon EC2 quando iniciada. Um perfil da instância pode conter somente uma função, e esse limite não pode ser aumentado. Se você criar a função usando o Console de gerenciamento da AWS, o perfil da instância será criado com o mesmo nome que a função. Para obter mais informações sobre os perfis da instância, consulte Usar perfis de instância.

Para obter informações sobre como executar uma instância do EC2 com um perfil, consulte Controle de acesso aos recursos do Amazon EC2 no Guia do usuário do Amazon EC2.

Para criar um perfil de instância e armazenar a função nele (AWS CLI)

1. Criar um perfil da instância: aws iam create-instance-profile
2. Adicionar a função ao perfil da instância: aws iam add-role-to-instance-profile

O comando da AWS CLI de exemplo definido a seguir demonstra as duas primeiras etapas para criar uma função e anexar permissões. Também mostra as duas etapas para criar um perfil de instância e adicionar a função ao perfil. Este exemplo de política de confiança permite que o serviço Amazon EC2 assuma a função e visualize o bucket `example_bucket` do Amazon S3. O exemplo também presume que você está executando em um computador cliente com o Windows e já configurou a interface de linha de comando com as credenciais de sua conta e região. Para obter mais informações, consulte Configurar a interface de linha de comando da AWS.

Neste exemplo, inclua a seguinte política de confiança no primeiro comando ao criar a função. Essa política de confiança permite que o serviço Amazon EC2 assuma a função.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Quando usa o segundo comando, você deve anexar uma política de permissões à função. A política de permissões de exemplo a seguir permite que a função execute apenas a ação `ListBucket` no bucket `example_bucket` do Amazon S3.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Para criar essa função `Test-Role-for-EC2`, você deve primeiro salvar a política de confiança anterior com o nome `trustpolicyforec2.json` e a política de permissões anterior com o nome `permissionspolicyforec2.json` no diretório `policies` na unidade `C:` local. Em seguida, você pode usar os seguintes comandos para criar a função, anexar a política, criar o perfil da instância e adicionar a função ao perfil da instância.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Quando você ativar a instância do EC2, especifique o nome do perfil da instância na página Configure Instance Details se você usar o console da AWS. Se você usar o comando CLI `aws ec2 run-instances`, especifique o parâmetro `--iam-instance-profile`.

Criar uma função para um serviço (API da AWS)

A criação de uma função na API da AWS envolve várias etapas. Quando o console é usado para criar uma função, muitas das etapas são concluídas por você, mas com a API, é necessário executar explicitamente cada etapa. Você deve criar a função e atribuir uma política de permissões à função. Se o serviço com o qual você está trabalhando for o Amazon EC2, você também deverá criar um

perfil de instância e adicionar a função a ele. Opcionalmente, você também pode definir o limite de permissões para sua função.

Para criar uma função para um serviço da AWS (API da AWS)

1. Criar uma função: CreateRole

   Para a política de confiança da função, você pode especificar um local de arquivo.
2. Anexar uma política de permissões gerenciada à função: AttachRolePolicy

   ou

   Criar uma política de permissões em linha para a função: PutRolePolicy
3. (Opcional) Adicione atributos personalizados ao perfil anexando tags: TagRole

   Para obter mais informações, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).
4. (Opcional) Definir o limite de permissões para a função: PutRolePermissionsBoundary

   Um limite de permissões controla o número máximo de permissões que uma função pode ter. Os limites de permissões são um recurso avançado da AWS.

Se você for usar a função com o Amazon EC2 ou outro produto da AWS que use o Amazon EC2, armazene a função em um perfil de instância. Um perfil da instância é um contêiner para uma função. Cada perfil da instância pode conter somente uma função e esse limite não pode ser aumentado. Se você criar a função no Console de gerenciamento da AWS, o perfil da instância será criado para você com o mesmo nome da função. Para obter mais informações sobre os perfis da instância, consulte Usar perfis de instância. Para obter informações sobre como executar uma instância do Amazon EC2 com um perfil, consulte Controle de acesso aos recursos do Amazon EC2 no Guia do usuário do Amazon EC2.

Para criar um perfil de instância e armazenar a função nele (API da AWS)

1. Criar um perfil da instância: CreateInstanceProfile
2. Adicionar a função ao perfil da instância: AddRoleToInstanceProfile

Criar um perfil vinculado ao serviço

Uma função vinculada ao serviço é um tipo exclusivo de função do IAM vinculado diretamente a um serviço da AWS. As funções vinculadas a serviços são predefinidas pelo serviço e incluem todas as permissões que o serviço requer para chamar outros serviços da AWS em seu nome. O serviço vinculado também define como criar, modificar e excluir uma função vinculada a serviço. Um serviço pode criar ou excluir a função automaticamente. Ele pode permitir que você crie, modifique ou exclua a função como parte de um assistente ou processo no serviço. Ou pode exigir que você use o IAM para criar ou excluir a função. Independente do método, as funções vinculadas a serviço simplificam o processo de configurar um serviço da , pois você não precisa adicionar manualmente as permissões para o serviço concluir as ações em seu nome.

> **Note**
>
> Lembre-se de que perfis de serviço são diferentes de perfis vinculados a serviços. O perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criação de um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM. Um perfil vinculado a serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode presumir o perfil de executar uma ação em seu nome. Perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.

O serviço vinculado define as permissões de suas funções vinculadas ao serviço e, a menos que definido em contrário, somente aquele serviço pode assumir as funções. As permissões definidas incluem a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

Para poder excluir as funções, você deve primeiroexcluir os recursos relacionados a eles. Isso ajuda a impedir que você remova por engano a permissão para acessar os recursos.

> **Tip**
>
> Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm

Sim na coluna Função vinculada ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Permissões de perfil vinculado ao serviço

Configure as permissões para que uma entidade do IAM (usuário ou função) permita que o usuário ou a função crie ou edite a função vinculada ao serviço.

> **Note**
>
> O ARN de uma função vinculada ao serviço inclui uma entidade principal do serviço que é indicada nas políticas abaixo como *SERVICE-NAME*.amazonaws.com. Não tente adivinhar a entidade principal do serviço, pois ela faz distinção entre maiúsculas e minúsculas, e o formato pode variar entre os serviços da AWS. Para visualizar a entidade principal do serviço, consulte a documentação do perfil vinculado ao serviço.

Para permitir que uma entidade do IAM; crie uma função vinculada ao serviço

Adicione a seguinte política à entidade do IAM que precisa criar a função vinculada ao serviço.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
```

```
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

Para permitir que uma entidade do IAM crie qualquer função vinculada ao serviço

Adicione a seguinte instrução à política de permissões da entidade do IAM que precisa criar uma função vinculada ao serviço ou qualquer função de serviço que inclua as políticas necessárias. Esta declaração de política não permite que a entidade do IAM anexe uma política à função.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Para permitir que uma entidade do IAM edite a descrição de todas as funções de serviço

Adicione a seguinte instrução à política de permissões da entidade do IAM que precisa editar uma descrição de uma função vinculada ao serviço ou qualquer função de serviço.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Para permitir que uma entidade do IAM exclua uma função vinculada ao serviço específica

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa excluir a função vinculada ao serviço.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
```

```
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

Para permitir que uma entidade do IAM exclua qualquer função vinculada ao serviço

Adicione a seguinte instrução à política de permissões da entidade do IAM que precisa excluir uma função vinculada ao serviço, mas não a função de serviço.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Para permitir que uma entidade do IAM transmita uma função existente para o serviço

Alguns serviços do AWS permitem que você transmita uma função existente para o serviço, ao invés de criar uma nova função vinculada ao serviço. Para fazer isso, um usuário deve ter permissões para transmitir a função para o serviço. Adicione a seguinte instrução à política de permissões para a entidade do IAM necessária para transmitir a função. Esta declaração de política também permite que a entidade visualize uma lista de funções a partir da qual elas podem escolher a função a ser transmitida. Para obter mais informações, consulte Conceda permissões a um usuário para passar um perfil para um serviço da AWS.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Permissões indiretas com funções vinculadas a serviços

As permissões concedidas por uma função vinculada a um serviço podem ser transferidas indiretamente para outros usuários e funções. Quando um perfil vinculado ao serviço é usado por um serviço da AWS, esse perfil vinculado ao serviço pode usar suas próprias permissões para chamar outros serviços da AWS. Isso significa que usuários e funções com permissões para chamar um serviço que usa uma função vinculada ao serviço podem ter acesso indireto aos serviços acessíveis por essa função vinculada ao serviço.

Por exemplo, quando você cria uma instância de banco de dados do Amazon RDS, uma função vinculada ao serviço para o RDS será criada automaticamente se ainda não houver uma. Essa função permite que o RDS chame o Amazon EC2, o Amazon SNS, o Amazon CloudWatch Logs e o Amazon Kinesis em seu nome. Se você permitir que usuários e perfis em sua conta modifiquem ou criem bancos de dados do RDS, eles poderão interagir indiretamente com o Amazon EC2, o Amazon SNS, os logs do Amazon CloudWatch Logs e os recursos do Amazon Kinesis por meio de chamadas ao RDS, pois o RDS usaria seu perfil vinculado ao serviço para acessar esses recursos.

Métodos de criação de perfis vinculados ao serviço

O método que você usa para criar uma função vinculada ao serviço depende do serviço. Em alguns casos, você não precisa criar manualmente uma função vinculada ao serviço. Por exemplo, quando você conclui uma ação específica (como a criação de um recurso) no serviço, o serviço pode criar a função vinculada ao serviço para você. Ou, se você estava usando um serviço antes que ele começou a oferecer suporte a funções vinculadas ao serviço, o serviço pode ter criado automaticamente a função na sua conta. Para saber mais, consulte Uma nova função apareceu na minha conta da AWS.

Em outros casos, o serviço pode oferecer suporte à criação de uma função vinculada ao serviço manualmente usando o console, API ou CLI do serviço. Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Para saber se o serviço oferece suporte para criar a função vinculada ao serviço, escolha o link Sim para visualizar a documentação da função vinculada a esse serviço.

Se o serviço não for compatível com a criação da função, você poderá usar o IAM para criar a função vinculada ao serviço.

> **Important**
>
> Perfis vinculados a serviço contam para o limite dos seus perfis do IAM em uma Conta da AWS, mas, se você tiver atingido seu limite, ainda poderá criar perfis vinculados a serviço na sua conta. Somente as funções vinculadas ao serviço podem exceder o limite.

Criar uma função vinculada ao serviço (console)

Antes de criar uma função vinculada ao serviço no IAM, descubra se o serviço vinculado cria automaticamente funções vinculadas ao serviço. Além disso, saiba se você pode criar a função no console, na API ou na CLI do serviço.

Para criar uma função vinculada ao serviço (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis. Depois, escolha Create role (Criar perfil).
3. Escolha o tipo de função AWS Service (Service).
4. Escolha o caso de uso para o seu serviço. Casos de uso são definidos pelo serviço para incluir a política de confiança exigida pelo serviço. Em seguida, escolha Próximo.
5. Escolha uma ou mais políticas de permissões a serem anexadas à função. Dependendo do caso de uso que você selecionou, o serviço pode executar uma das seguintes ações:

   - Defina as permissões usadas pela função.
   - Permita que você escolha um conjunto limitado de permissões.
   - Permita que você escolha entre todas as permissões.
   - Permitir que você opte por não selecionar nenhuma política no momento, criar políticas mais tarde e, em seguida, anexá-las à função.

   Marque a caixa de seleção ao lado da política que atribui as permissões que você deseja que o perfil tenha e escolha Próximo.

> **Note**
>
> As permissões que você especificar estarão disponíveis para qualquer entidade que usar a função. Por padrão, uma função não tem nenhuma permissões.

6. Para Nome do perfil, o grau de personalização do nome do perfil é definido pelo serviço. Se o serviço define o nome da função, essa opção não é editável. Em outros casos, o serviço pode definir um prefixo para a função e permitir que você insira um sufixo opcional.

   Se possível, insira um sufixo de nome de função a ser adicionado ao nome padrão. O sufixo ajuda a identificar a finalidade dessa função. Os nomes de função devem ser exclusivos em sua conta AWS. Eles não são diferenciados por letras maiúsculas e minúsculas. Por exemplo, não é possível criar perfis denominados **`<service-linked-role-name>_SAMPLE`** e **`<service-linked-role-name>_sample`**. Como várias entidades podem fazer referência à função, não é possível editar o nome da função depois que ela é criada.
7. (Opcional) Em Description (Descrição), edite a descrição para a nova função vinculado ao serviço.
8. Você não pode anexar tags para funções vinculadas ao serviço durante a criação. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
9. Revise o perfil e escolha Criar perfil.

Criar uma função vinculada ao serviço (AWS CLI)

Antes de criar uma função vinculada ao serviço no IAM, descubra se o serviço vinculado cria automaticamente as funções vinculadas ao serviço e se você pode criar a função na CLI do serviço. Se a CLI do serviço não for compatível, você poderá usar comandos do IAM para criar uma função vinculada ao serviço com a política de confiança e políticas em linha de que o serviço precisa assumir a função.

Para criar uma função vinculada ao serviço (AWS CLI)

Execute o seguinte comando:

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Criar uma função vinculada ao serviço (API da AWS)

Antes de criar uma função vinculada ao serviço no IAM, descubra se o serviço vinculado cria automaticamente as funções vinculadas ao serviço e se você pode criar a função na API do serviço. Se a API do serviço não for suportada, você pode usar a API do AWS para criar uma função vinculada ao serviço com a política de confiança e políticas em linha de que o serviço precisa para assumir a função.

Para criar uma função vinculada ao serviço (API da AWS)

Use a chamada de API CreateServiceLinkedRole. Na solicitação, especifique o nome do serviço na forma de *SERVICE_NAME_URL*`.amazonaws.com`.

Por exemplo, para criar a função vinculada ao serviço Lex Bots, use `lex.amazonaws.com`.

Criar um perfil para um provedor de identidade de terceiros

Você pode usar provedores de identidade em vez de criar usuários do IAM na sua Conta da AWS. Com um provedor de identidade (IdP), você pode gerenciar as identidades de usuários fora da AWS e fornecer a essas identidades de usuários externos permissões para acessar recursos da AWS na sua conta. Para obter mais informações sobre federação e provedores de identidade, consulte Federação e provedores de identidade na AWS.

Criar um perfil para entidades principais federadas OIDC e SAML (console)

Os procedimentos para criar um perfil dependem de sua opção de provedores de terceiros:

- Para OpenID Connect (OIDC), consulte Criar uma função para uma federação do OpenID Connect (console).
- Para SAML 2.0, consulte Criar um perfil para uma federação do SAML 2.0 (console).

Criar uma função para acesso federado (AWS CLI)

As etapas para criar uma função para provedores de identidade (OIDC ou SAML) na AWS CLI são idênticas. A diferença está no conteúdo da política de confiança que você cria nas etapas obrigatórias. Comece seguindo as etapas na seção Pré-requisitos para o tipo de provedor que você está usando:

- Para um provedor do OIDC, consulte Pré-requisitos para a criação de uma função para o OIDC.
- Para um provedor do SAML, consulte Pré-requisitos para a criação de uma função para o SAML.

A criação de uma função a partir da AWS CLI envolve várias etapas. Quando o console é usado para criar uma função, muitas das etapas são concluídas por você, mas com a AWS CLI, é necessário executar explicitamente cada etapa. Você deve criar a função e atribuir uma política de permissões à função. Opcionalmente, você também pode definir o limite de permissões para sua função.

Para criar um perfil (AWS CLI)

1. Criar uma função: aws iam create-role
2. Anexar uma política de permissões à função: aws iam attach-role-policy

   ou

   Criar uma política de permissões em linha para a função: aws iam put-role-policy
3. (Opcional) Adicione atributos personalizados à função anexando tags: aws iam tag-role

   Para obter mais informações, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).
4. (Opcional) Definir o limite de permissões para a função: aws iam put-role-permissions-boundary

   Um limite de permissões controla o número máximo de permissões que uma função pode ter. Os limites de permissões são um recurso avançado da AWS.

O exemplo a seguir mostra as duas primeiras e as etapas mais comuns para a criação de uma função de provedor de identidade em um ambiente simples. Este exemplo permite que qualquer usuário na conta `123456789012` assuma a função e visualize o bucket `example_bucket` do Amazon S3. Este exemplo também supõe que você esteja executando a AWS CLI em um computador Windows e já tenha configurado a AWS CLI com suas credenciais. Para obter mais informações, consulte Configurar a AWS Command Line Interface.

A política de confiança do exemplo a seguir será designada para um aplicativo móvel se o usuário fizer login usando o Amazon Cognito. Neste exemplo, *`us-east:12345678-ffff-ffff-ffff-123456`* representa o ID do grupo de identidades atribuído pelo Amazon Cognito.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
```

```
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

A política de permissões a seguir permite que qualquer pessoa que assuma a função execute apenas a ação `ListBucket` no bucket `example_bucket` do Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Para criar essa função `Test-Cognito-Role`, você deve primeiro salvar a política de confiança anterior com o nome `trustpolicyforcognitofederation.json` e a política de permissões anterior com o nome `permspolicyforcognitofederation.json` na pasta `policies` na unidade `C:` local. Você pode usar os comandos a seguir para criar a função e anexar a política em linha.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json

# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Criar uma função para acesso federado (API da AWS)

As etapas para criar uma função para provedores de identidade (OIDC ou SAML) na AWS CLI são idênticas. A diferença está no conteúdo da política de confiança que você cria nas etapas obrigatórias. Comece seguindo as etapas na seção Pré-requisitos para o tipo de provedor que você está usando:

- Para um provedor do OIDC, consulte Pré-requisitos para a criação de uma função para o OIDC.
- Para um provedor do SAML, consulte Pré-requisitos para a criação de uma função para o SAML.

Para criar um perfil (API AWS)

1. Criar uma função: CreateRole
2. Anexar uma política de permissões à função: AttachRolePolicy

   ou

   Criar uma política de permissões em linha para a função: PutRolePolicy
3. (Opcional) Adicione atributos personalizados ao perfil anexando tags: TagRole

   Para obter mais informações, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).
4. (Opcional) Definir o limite de permissões para a função: PutRolePermissionsBoundary

   Um limite de permissões controla o número máximo de permissões que uma função pode ter. Os limites de permissões são um recurso avançado da AWS.

Criar uma função para uma federação do OpenID Connect (console)

É possível usar os provedores de identidades federadas do OpenID Connect (OIDC) em vez de criar usuários do AWS Identity and Access Management em sua Conta da AWS. Com um provedor de identidade (IdP), você pode gerenciar as identidades de usuários fora da AWS e fornecer a essas identidades de usuários externos permissões para acessar recursos da AWS na sua conta. Para obter mais informações sobre federação e IdPs, consulte Federação e provedores de identidade na AWS.

Pré-requisitos para a criação de uma função para o OIDC

Para criar uma função para a federação OIDC, você primeiro deve concluir as etapas obrigatórias a seguir.

Para se preparar para uma função para a federação do OIDC

1. Inscreva-se em um ou mais serviços que oferecem identidades federadas OIDC. Se estiver criando um aplicativo que precisa de acesso a seus recursos da AWS, você também poderá configurar seu aplicativo com as informações do provedor. Quando você faz isso, o provedor fornece a você um ID de aplicação ou público que é exclusivo da aplicação. (Diferentes provedores usam diferentes terminologias para este processo. Este guia usa o termo configurar para o processo de identificação de sua aplicação com o provedor.) É possível configurar várias aplicações com cada provedor ou vários provedores com um único aplicativo. Visualize as informações sobre o uso de provedores de identidades:

   - Centro do desenvolvedor do Login with Amazon
   - Adicione o login do Facebook ao seu aplicativo ou site no site de desenvolvedores do Facebook.
   - Uso de OAuth 2.0 para login (OpenID Connect) no site de desenvolvedores do Google.
2. Depois de receber as informações necessárias do IdP, crie um IdP no IAM. Para obter mais informações, consulte Criar um provedor de identidade OpenID Connect (OIDC) no IAM.

   > **Important**
   >
   > Se você estiver usando um IdP OIDC do Google, Facebook ou Amazon Cognito, não crie um IdP do IAM separado no Console de gerenciamento da AWS. Esses provedores de identidades OIDC já estão integrados à AWS e estão disponíveis para uso. Ignore esta etapa e a criação de novas funções usando seu IdP na etapa a seguir.
3. Prepare as políticas para a função que os usuários autenticados pelo IdP assumirão. Assim como com qualquer função, uma função para um aplicativo móvel inclui duas políticas. Uma é a política de confiança que especifica quem pode assumir a função. A outra é a política de permissões que especifica as ações e os recursos reais da AWS aos quais o aplicativo móvel tem ou não permissão para acessar.

   Para idPs da Web, recomendamos que você use o Amazon Cognito para gerenciar identidades. Nesse caso, use uma política de confiança semelhante a este exemplo.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

Substitua `us-east-2:12345678-abcd-abcd-abcd-123456` pelo ID do grupo de identidades que o Amazon Cognito atribuir a você.

Se você configurar manualmente um IdP do OIDC, ao criar a política de confiança, você deve usar três valores que garantem que apenas seu aplicativo possa assumir a função:

- No elemento `Action`, use a ação `sts:AssumeRoleWithWebIdentity`.
- No elemento `Principal`, use a string `{"Federated":`*`providerUrl/providerArn`*`}`.
  - Para alguns IdPs OIDC comuns, a *`providerUrl`* é um URL. Os exemplos a seguir incluem métodos para especificar a entidade principal para alguns IdPs comuns:

    `"Principal":{"Federated":"cognito-identity.amazonaws.com"}`

    `"Principal":{"Federated":"www.amazon.com"}`

    `"Principal":{"Federated":"graph.facebook.com"}`

    `"Principal":{"Federated":"accounts.google.com"}`
  - Para outros provedores OIDC, use nome do recurso da Amazon (ARN) do IdP OIDC que você criou em Step 2, como o exemplo a seguir:

```
"Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
server.example.com"}
```

- No elemento `Condition`, use uma condição `StringEquals` para limitar as permissões. Teste o ID do grupo de identidades para o Amazon Cognito ou o ID da aplicação para outros provedores. O ID do grupo de identidades deve corresponder ao ID da aplicação que você recebeu quando configurou a aplicação com o IdP. Essa correspondência entre os IDs garante que a solicitação seja proveniente da aplicação.

> **Note**
>
> Os perfis do IAM para bancos de identidades do Amazon Cognito confiam na entidade principal do serviço `cognito-identity.amazonaws.com` para assumir o perfil. Perfis desse tipo devem conter pelo menos uma chave de condição para limitar as entidades principais que podem assumir o perfil.
> Considerações adicionais se aplicam aos bancos de identidades do Amazon Cognito que assumem perfis do IAM entre contas. As políticas de confiança desses perfis devem aceitar a entidade principal do serviço `cognito-identity.amazonaws.com` e conter a chave de condição `aud` para restringir a suposição de perfis aos usuários dos bancos de identidades pretendidos. Uma política que confia nos bancos de identidades do Amazon Cognito sem essa condição cria o risco de que um usuário de um banco de identidades não intencional possa assumir o perfil. Para obter mais informações, consulte Políticas de confiança para perfis do IAM na autenticação básica (clássica) no Guia do Desenvolvedor do Amazon Cognito.

Crie um elemento de condição semelhante aos exemplos a seguir, dependendo do IdP que você está usando:

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

Para provedores OIDC, use o URL totalmente qualificado do IdP OIDC com a chave de contexto aud, como o exemplo a seguir:

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> Observe que os valores para a entidade principal na política de confiança para a função são específicos a um IdP. Um perfil pode especificar apenas uma entidade principal. Portanto, se a aplicação móvel permitir que os usuários se registrem em mais de um IdP, você deverá criar uma função separada para cada IdP que utilizar. Portanto, você deve criar políticas de confiança separadas para cada IdP.

Se um usuário usar uma aplicação móvel para fazer login estando no Login with Amazon, a política de confiança do exemplo a seguir será aplicada. No exemplo, *amzn1.application-oa2-123456* representa o ID da aplicação que a Amazon atribuiu quando você configurou a aplicação usando o Login with Amazon.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Se um usuário usar uma aplicação móvel para fazer login estando no Facebook, a política de confiança do exemplo a seguir será aplicada. Neste exemplo, *111222333444555* representa o ID da aplicação atribuído pelo Facebook.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Se um usuário usar uma aplicação móvel para fazer login estando no Goggle, a política de confiança do exemplo a seguir será aplicada. Nesse exemplo, *666777888999000* representa o ID da aplicação atribuído pelo Google.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Se um usuário usar uma aplicação móvel para fazer login estando no Amazon Cognito, a política de confiança do exemplo a seguir será aplicada. Neste exemplo, *us-east:12345678-ffff-ffff-ffff-123456* representa o ID do grupo de identidades atribuído pelo Amazon Cognito.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-
identity.amazonaws.com:aud": "us-east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Como criar um perfil para o OIDC

Depois de concluir as etapas obrigatórias, você pode criar a função no IAM. Para provedores de identidade (IdPs) OpenID Connect (OIDC) compartilhados reconhecidos, o IAM exige uma avaliação explícita de declarações específicas em JSON Web Tokens (JWTs), os quais são conhecidos como controles de provedor de identidade. Para obter mais informações sobre quais IdPs OIDC têm controles de provedor de identidade, consulte Controles de provedor de identidade para provedores OIDC compartilhados.

O procedimento a seguir descreve como criar o perfil para federação OIDC no Console de gerenciamento da AWS. Para criar uma função na AWS CLI ou na API da AWS, consulte os procedimentos em Criar um perfil para um provedor de identidade de terceiros.

> **Important**
>
> Se você estiver usando o Amazon Cognito, use o console do Amazon Cognito para configurar as funções. Caso contrário, use o console do IAM para criar uma função para federação OIDC.

Criar um perfil do IAM para a federação OIDC

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Funções e Criar função.
3. Escolha a identidade da Web como o tipo de entidade confiável e selecione Avançar.
4. Em Identity provider (Provedor de identidades), escolha o IdP para a função:

   - Se você desejar criar uma função para um IdP da Web individual, selecione Login with Amazon, Facebook ou Google.

     > **Note**
     >
     > Você deve criar uma função separada para cada IdP que quiser utilizar.

   - Se você desejar criar uma função de cenário avançado para o Amazon Cognito, escolha Amazon Cognito.

     > **Note**
     >
     > Você só precisa criar manualmente uma função para usar com o Amazon Cognito quando está trabalhando em um cenário avançado. Caso contrário, o Amazon Cognito pode criar funções para você. Para obter mais informações sobre o Amazon Cognito, consulte Provedores externos de identidade de grupos de identidades (identidades federadas) no Guia do desenvolvedor do Amazon Cognito.

   - Para criar uma função para o GitHub Actions, é necessário adicionar o provedor OIDC do GitHub ao IAM. Depois de adicionar o provedor OIDC do GitHub ao IAM, escolha token.actions.githubusercontent.com.

     > **Note**
     >
     > Para obter informações sobre como configurar a AWS para confiar no provedor de OIDC do GitHub como uma identidade federada, consulte Documentos do GitHub – Configuração do OpenID Connect na Amazon Web Services. Para obter informações sobre as práticas recomendadas para limitar o acesso às funções associadas ao IAM

IdP para GitHub, consulte Configurar uma função para o provedor de identidades OIDC GitHub nesta página.

- Se você desejar criar um perfil para a HashiCorp Cloud Platform (HCP) Terraform, comece adicionando o provedor de OIDC do Terraform ao IAM. Depois de adicionar o provedor de OIDC Terraform ao IAM, escolha app.terraform.io.

> **Important**
>
> Os perfis do IAM para o provedor de OIDC Terraform da HashiCorp Cloud Platform (HCP) devem avaliar a chave de condição do IAM, `app.terraform.io:sub`, na política de confiança do perfil. Essa chave de condição limita quais organizações, projetos, espaços de trabalho ou fases de execução do HCP Terraform são capazes de assumir o perfil. Sem essa chave de condição, sua política de confiança concede acesso ao seu perfil e a recursos da AWS por meio de identidades fora da sua organização, o que não se alinha ao princípio de privilégio mínimo.
> Se você definir ou modificar uma política de confiança de um perfil para um perfil associado ao provedor de OIDC Terraform da HCP em sua conta da AWS, mas não avaliar a chave de condição do IAM `app.terraform.io:sub`, você receberá um erro. Além disso, o AWS STS negará solicitações de autorização se sua política de confiança de perfil não avaliar essa chave de condição.

5. As informações solicitadas variam de acordo com o provedor OIDC que você escolher.

   - Insira o identificador para a aplicação. O rótulo do identificador é alterado de acordo com o provedor escolhido:
     - Se você desejar criar uma função para o Login with Amazon, insira o ID da aplicação na caixa Application ID (ID da aplicação).
     - Se você desejar criar uma função para o Facebook, digite o ID da aplicação na caixa Application ID (ID da aplicação).
     - Se você desejar criar uma função para o Google, digite o nome do público na caixa Audience (Público).
     - Se você desejar criar uma função para o Amazon Cognito, digite o ID do grupo de identidades que você criou para as aplicações do Amazon Cognito na caixa Identity Pool ID (ID do grupo de identidades).
   - Para criar uma função para GitHub Actions, insira os seguintes detalhes:

- Em Público, escolha `sts.amazonaws.com`.
- Em Organização do GitHub, insira o nome da organização no GitHub. O nome da organização do GitHub é obrigatório e deve ser alfanumérico, incluindo traços (-). Não é permitido usar caracteres curinga (* e ?) no nome da organização do GitHub.
- (Opcional) Em Repositório do GitHub, insira o URL do seu repositório do GitHub. Se você não especificar um valor, o padrão será um coringa (*).
- (Opcional) em Filial do GitHub, insira o nome da filial do GitHub. Se você não especificar um valor, o padrão será um coringa (*).

- Se você desejar criar um perfil para a HashiCorp Cloud Platform (HCP) Terraform, insira os detalhes a seguir:
  - Em Público, escolha `aws.workload.identity`.
  - Em Organização, insira o nome da organização. É possível especificar um caractere curinga (*) para todas as organizações.
  - Em Projeto, insira o nome do projeto. É possível especificar um caractere curinga (*) para todos os projetos.
  - Em Workspace, insira o nome do workspace. É possível especificar um caractere curinga (*) para todos os workspaces.
  - Em Fase de execução, insira o nome da fase de execução. É possível especificar um caractere curinga (*) para todas as fases de execução.

6. (Opcional) em Condição (opcional), escolha Adicionar condição para criar condições adicionais que devem ser atendidas para que os usuários da aplicação possam usar as permissões concedidas pela função. Por exemplo, você pode adicionar uma condição que conceda acesso a recursos da AWS apenas para um determinado ID de usuário do IAM. Você também pode adicionar condições à política de confiança após a criação da função. Para obter mais informações, consulte Atualizar a política de confiança de um perfil.
7. Revise suas informações de OIDC e escolha Próximo.
8. O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta. Selecione a política a ser usada para a política de permissões ou escolha Create policy (Criar política) para abrir uma nova guia no navegador e criar uma nova política a partir do zero. Para obter mais informações, consulte Criação de políticas do IAM. Depois de criar a política, feche essa guia e retorne à guia original. Marque a caixa de seleção ao lado das políticas de permissões que você deseja que os usuários do OIDC tenham. Se preferir, você pode optar por não selecionar nenhuma política neste momento e anexar as políticas à função mais tarde. Por padrão, uma função não tem nenhuma permissões.

9. (Opcional) Defina um limite de permissões. Este é um recurso avançado.

   Abra a seção Set permissions boundary (Definir limite de permissões) e escolha Use a permissions boundary to control the maximum role permissions (Usar um limite de permissões para controlar o número máximo de permissões de função). Selecione a política a ser usada para o limite de permissões.
10. Escolha Próximo.
11. Em Role name (Nome da função), insira um nome. Os nomes de função devem ser exclusivos em sua Conta da AWS. Eles não diferenciam maiúsculas e minúsculas. Por exemplo, não é possível criar duas funções denominadas **`PRODROLE`** e **`prodrole`**. Como outros recursos da AWS podem referenciar a função, não é possível editar o nome da função depois que ele é criado.
12. (Opcional) Em Description (Descrição), insira uma descrição para a nova função.
13. Para editar os casos de uso e as permissões da função, escolha Edit (Editar) nas seções Etapa 1: selecionar entidades confiáveis ou na Etapa 2: adicionar permissões.
14. (Opcional) Para adicionar metadados à função, anexe tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
15. Revise o perfil e escolha Criar perfil.

Configurar uma função para o provedor de identidades OIDC GitHub

Se você usar o GitHub como um provedor de identidades (IdP) Open ID Connect (OIDC), a prática recomendada será limitar as entidades que podem assumir o perfil associado ao IdP do IAM. Ao incluir uma instrução de condição na política de confiança, você pode limitar a função a uma organização, repositório ou ramificação específica do GitHub. É possível usar a chave de condição `token.actions.githubusercontent.com:sub` com operadores de condição de string para limitar o acesso. Recomendamos limitar a condição a um conjunto específico de repositórios ou ramificações em sua organização do GitHub. Para obter informações sobre como configurar a AWS para confiar no OIDC do GitHub como uma identidade federada, consulte GitHub Docs - Configuring OpenID Connect in Amazon Web Services ("Documentos do GitHub - Configurar o OpenID Connect na Amazon Web Services").

Se você usar ambientes do GitHub em fluxos de trabalho de ação ou em políticas de OIDC, é extremamente recomendável adicionar regras de proteção ao ambiente para aumentar a segurança. Use ramificações e tags de implantação para restringir quais ramificações e tags podem ser

implantadas no ambiente. Para obter mais informações sobre a configuração de ambientes com regras de proteção, consulte Ramificações de implantação e marcas no artigo Usando ambientes para implantação do GitHub.

Quando o IdP OIDC do GitHub é a entidade principal confiável para seu perfil, o IAM verifica a condição da política de confiança do perfil para verificar se a chave de condição `token.actions.githubusercontent.com:sub` está presente e se seu valor não é apenas um caractere curinga (* e ?) ou nulo. O IAM realiza essa verificação quando a política de confiança é criada ou atualizada. Se a chave de condição `token.actions.githubusercontent.com:sub` não estiver presente ou o valor da chave não satisfizer os critérios de valor mencionados, a solicitação falhará e retornará um erro.

> **Important**
>
> Se você não limitar a chave de condição `token.actions.githubusercontent.com:sub` a uma organização ou um repositório específico, as ações do GitHub de organizações ou repositórios fora do seu controle poderão assumir perfis associados ao IdP do IAM do GitHub na sua conta da AWS.

O exemplo de política de confiança a seguir limita o acesso à organização, repositório e ramificação definidos do GitHub. O valor da chave de condição `token.actions.githubusercontent.com:sub` no exemplo a seguir é o formato padrão do valor do assunto documentado pelo GitHub.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
```

```
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

O exemplo de condição a seguir limita o acesso à organização e ao repositório definidos do GitHub, mas concede acesso a qualquer ramificação dentro do repositório.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

O exemplo de condição a seguir limita o acesso a qualquer repositório ou ramificação dentro da organização do GitHub definida. Recomendamos limitar a chave de condição `token.actions.githubusercontent.com:sub` a um valor específico que limita o acesso ao GitHub Actions de dentro da sua organização do GitHub.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
}
```

Para obter mais informações sobre chaves de federação OIDC disponíveis para verificação de condições nas políticas, consulte Chaves disponíveis para federação OIDC da AWS.

Criar um perfil para uma federação do SAML 2.0 (console)

Você pode usar a federação do SAML 2.0 em vez de criar usuários do IAM na sua Conta da AWS. Com um provedor de identidade (IdP), você pode gerenciar as identidades de usuários fora da AWS e fornecer a essas identidades de usuários externos permissões para acessar recursos da AWS na sua conta. Para obter mais informações sobre federação e provedores de identidade, consulte Federação e provedores de identidade na AWS.

> **Note**
>
> Para melhorar a resiliência da federação, recomendamos que você configure seu IdP e sua federação da AWS para oferecer suporte a vários endpoints de login do SAML. Para obter detalhes, consulte o artigo do AWS Security Blog How to use regional SAML endpoints for failover.

Pré-requisitos para a criação de uma função para o SAML

Para criar uma função para a federação SAML 2.0, você primeiro deve concluir as etapas obrigatórias a seguir.

Para se preparar para uma função para a federação do SAML 2.0

1. Antes de criar uma função para federação baseada em SAML, você deve criar um provedor SAML no IAM. Para obter mais informações, consulte Criar um provedor de identidades SAML no IAM.
2. Prepare as políticas para a função que os usuários autenticados pelo SAML 2.0 assumirão. Assim como com qualquer função, uma função para a federação do SAML inclui duas políticas. Uma é a política de confiança da função que especifica quem pode assumir a função. A outra é a política de permissões do IAM que especifica as ações e os recursos da AWS aos quais a entidade principal federada SAML tem ou não permissão para acessar.

   Quando você criar a política de confiança para sua função, deverá usar três valores para garantir que a função só possa ser assumida por sua aplicação:

   - No elemento `Action`, use a ação `sts:AssumeRoleWithSAML`.
   - No elemento `Principal`, use a string `{"Federated":`*`ARNofIdentityProvider`*`}`. Substitua *`ARNofIdentityProvider`* pelo nome de recurso da Amazon (ARN) do provedor de identidade SAML que você criou em Step 1.

- Para o elemento `Condition`, use uma condição `StringEquals` para verificar se o atributo `saml:aud` da resposta SAML corresponde ao URL exibido pelo seu navegador durante o início de sessão no console. Essa URL do endpoint de início de sessão é o atributo de destinatário SAML do seu provedor de identidade. É possível incluir URLs de início de sessão em determinadas regiões. A AWS recomenda usar endpoints regionais em vez do endpoint global para melhorar a resiliência da federação. Para obter uma lista dos possíveis valores de *`region-code`* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS).

  Se a criptografia SAML for necessária, o URL de início de sessão deverá incluir o identificador exclusivo que a AWS atribui ao seu provedor SAML. É possível visualizar o identificador exclusivo selecionando o provedor de identidade no console do IAM para exibir a página de detalhes.

  `https://`*`region-code`*`.signin.aws.amazon.com/saml/acs/`*`IdP-ID`*

A política de confiança de exemplo a seguir foi projetada para um usuário federado do SAML:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRoleWithSAML",
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:saml-provider/PROVIDER-
NAME"
        },
        "Condition": {
            "StringEquals": {
                "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
            }
        }
    }
}
```

Substitua o ARN da entidade de segurança pelo ARN real do provedor SAML que você criou no IAM. Ele terá seu próprio ID de conta e nome de provedor.

Criar uma função para o SAML

Depois de concluir as etapas obrigatórias, você pode criar a função para federação baseada no SAML.

Para criar uma função para a federação baseada em SAML

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Roles (Funções) e Criar função (Create role).
3. Escolha o tipo de função Federação SAML 2.0.
4. Em SAML Provider (Provedor SAML), escolha o provedor para a função.
5. Escolha o método de nível de acesso SAML 2.0.

   - Escolha Allow programmatic access only (Permitir acesso programático apenas) para criar uma função que possa ser assumida programaticamente na API da AWS ou na AWS CLI.
   - Escolha Allow programmatic and Console de gerenciamento da AWS access (Permitir acesso programático e do console) para criar uma função que possa ser assumida de forma programática e no Console de gerenciamento da AWS.

   As funções criadas por ambos são semelhantes, mas a função que também pode ser assumida no console inclui uma política de confiança com uma determinada condição. Essa condição garante explicitamente que o público de SAML (atributo `SAML:aud`) seja definido como o endpoint de início de sessão da AWS para seu provedor SAML.
6. O procedimento para definir atributos varia em função do tipo de acesso.
   - Se estiver criando uma função para acesso programático, escolha um atributo na lista Atributo. Em seguida, na caixa Value (Valor), insira um valor a ser incluído na função. Isso restringe o acesso à função a usuários do provedor de identidade cuja resposta de autenticação SAML (declaração) inclua os atributos especificados. Você deve especificar pelo menos um atributo para garantir que sua função seja limitada a um subconjunto de usuários em sua organização.

- Se você estiver criando um perfil para acesso programático e via Console de gerenciamento da AWS, a seção Endpoints de início de sessão define o URL que seu navegador exibe durante o início de sessão no console. Esse endpoint é o atributo de destinatário SAML do seu provedor de identidade, que é mapeado na chave de contexto `saml:aud`. Para obter mais informações, consulte Configurar declarações SAML para a resposta de autenticação.

  a. Escolha Endpoints regionais ou Endpoint não regional. Recomendamos o uso de vários endpoints de início de sessão SAML regionais para melhorar a resiliência da federação.

  b. Em Regiões, escolha as regiões às quais seu provedor de SAML oferece suporte para início de sessão na AWS.

  c. Em URLs de início de sessão devem incluir identificadores exclusivos, selecione se os endpoints de início de sessão incluem os identificadores exclusivos que a AWS atribui ao seu provedor de identidade SAML. Essa opção é necessária para asserções SAML criptografadas. Para obter mais informações, consulte Federação SAML 2.0.

7. Para adicionar à política de confiança mais condições relacionadas a atributos, escolha Condition (optional) (Adicionar condições [opcional]), selecione a condição adicional e especifique um valor.

> **Note**
>
> A lista inclui os atributos do SAML usados com mais frequência. O IAM oferece suporte a atributos adicionais que você pode usar para criar condições. Para obter uma lista dos atributos com suporte, consulte Chaves disponíveis para a federação do SAML Se precisar de uma condição para um atributo do SAML compatível que não esteja na lista, você poderá adicionar essa condição manualmente. Para fazer isso, edite a política de confiança depois de criar a função.

8. Revise as informações de confiança do SAML 2.0 e, em seguida, escolha Next (Avançar).

9. O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta. Selecione a política a ser usada para a política de permissões ou escolha Create policy (Criar política) para abrir uma nova guia no navegador e criar uma nova política a partir do zero. Para obter mais informações, consulte Criação de políticas do IAM. Depois de criar a política, feche essa guia e retorne à guia original. Marque a caixa de seleção ao lado das políticas de permissões que deseja que os usuários federados SAML tenham. Se preferir, você pode optar por não selecionar nenhuma política neste momento e anexar as políticas à função mais tarde. Por padrão, uma função não tem nenhuma permissões.

10. (Opcional) Defina um limite de permissões. Este é um recurso avançado.

    Abra a seção Set permissions boundary (Definir limite de permissões) e escolha Use a permissions boundary to control the maximum role permissions (Usar um limite de permissões para controlar o número máximo de permissões de função). Selecione a política a ser usada para o limite de permissões.

11. Escolha Próximo.
12. Escolha Próximo: revisar.
13. Em Role name (Nome da função), insira um nome. Os nomes de função devem ser exclusivos em sua Conta da AWS. Eles não são diferenciados por letras maiúsculas e minúsculas. Por exemplo, não é possível criar perfis denominados **`PRODROLE`** e **`prodrole`**. Como outros recursos de AWS podem fazer referência à função, não é possível editar o nome da função depois de ela ser criada.
14. (Opcional) Em Description (Descrição), insira uma descrição para a nova função.
15. Escolha Edit (Editar) nas seções Etapa 1: selecionar entidades confiáveis ou na Etapa 2: adicionar permissões para editar os casos de uso e as permissões para a função.
16. (Opcional) Adicione metadados à função anexando tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
17. Revise o perfil e escolha Criar perfil.

Depois de criar a função, você conclui a confiança do SAML configurando o software do provedor de identidade com informações sobre a AWS. Essas informações incluem as funções que você deseja que seus usuários federados SAML usem. Isso é chamado de configuração da confiança da parte confiável entre seu IdP e a AWS. Para obter mais informações, consulte Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações.

Criar um perfil usando políticas de confiança personalizadas

Você pode criar uma política de confiança personalizada para delegar acesso e permitir que outras pessoas realizem ações na sua Conta da AWS. Para obter mais informações, consulte Criação de políticas do IAM.

Para obter informações sobre como usar funções para delegar permissões, consulte Termos e conceitos das funções.

Criar um perfil do IAM usando políticas de confiança personalizada (console)

Você pode usar o Console de gerenciamento da AWS para criar uma função que um usuário do IAM pode assumir. Por exemplo, suponha que sua organização tem várias Contas da AWS para isolar um ambiente de desenvolvimento de um ambiente de produção. Para obter informações de alto nível sobre a criação de uma função que permita que os usuários na conta de desenvolvimento acessem recursos na conta de produção, consulte Cenário de exemplo que usa contas separadas de desenvolvimento e produção.

Para criar uma função usando uma política de confiança personalizadas (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Roles (Perfis) e, em seguida, clique em Create role (Criar perfil).
3. Selecione o tipo de função Custom trust policy (Política de confiança personalizada).
4. Na seção Custom trust policy (Política de confiança personalizada), insira ou cole a política de confiança personalizada para a função. Para obter mais informações, consulte Criação de políticas do IAM.
5. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a validação de política e depois escolha Next (Avançar).
6. (Opcional) Defina um limite de permissões. Esse é um atributo avançado que está disponível para perfis de serviço, mas não para perfis vinculados ao serviço.

   Abra a seção Set permissions boundary (Definir limite de permissões) e escolha Use a permissions boundary to control the maximum role permissions (Usar um limite de permissões para controlar o número máximo de permissões de função). O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta. Selecione a política a ser usada para o limite de permissões.
7. Escolha Próximo.
8. Para Nome do perfil, o grau de personalização do nome do perfil é definido pelo serviço. Se o serviço definir o nome da função, essa opção não será editável. Em outros casos, o serviço pode definir um prefixo para a função e permitir que você informe um sufixo opcional. Alguns serviços permitem que você especifique todo o nome de sua função.

   Se possível, insira um nome de função ou um sufixo de nome de função. Os nomes de função devem ser exclusivos em sua Conta da AWS. Eles não são diferenciados por letras maiúsculas

e minúsculas. Por exemplo, não é possível criar perfis denominados **PRODROLE** e **prodrole**. Como outros recursos de AWS podem fazer referência à função, não é possível editar o nome da função depois de ela ser criada.

9. (Opcional) Em Description (Descrição), insira uma descrição para a nova função.
10. (Opcional) Escolha Editar nas seções Etapa 1: selecionar entidades confiáveis ou Etapa 2: adicionar permissões para editar a política personalizada e as permissões do perfil.
11. (Opcional) Adicione metadados à função anexando tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
12. Revise o perfil e escolha Criar perfil.

Exemplos de políticas para delegação de acesso

Os exemplos a seguir mostram como você pode permitir ou conceder acesso para uma Conta da AWS aos recursos em outra Conta da AWS. Para saber como criar uma política do IAM usando esses exemplos de documentos de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

Tópicos

- Uso de perfis para delegar acesso aos recursos de outra Conta da AWS
- Uso de uma política para delegar acesso a serviços
- Uso de uma política baseada em recurso para delegar acesso a um bucket do Amazon S3 em outra conta
- Uso de uma política baseada em recurso para delegar acesso a uma fila do Amazon SQS em outra conta
- Não é possível delegar acesso quando o acesso à conta é negado

Uso de perfis para delegar acesso aos recursos de outra Conta da AWS

Para obter um tutorial que mostra como usar funções do IAM para conceder aos usuários em uma conta acesso a recursos da AWS que estão em outra conta, consulte Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM.

> **Important**
>
> Você pode incluir o ARN de uma função ou usuário específico no elemento `Principal` de uma política de confiança de função. Quando você salva a política, o AWS transforma o ARN em um ID principal exclusivo. Isso ajuda a reduzir o risco de alguém elevar seus privilégios ao remover e recriar a função ou usuário. Normalmente, você não vê esse ID no console, porque há também uma transformação reversa de volta para o ARN quando a política de confiança é exibida. No entanto, se você excluir a função ou usuário, o relacionamento é interrompido. A política não se aplica mais, mesmo se você recriar o usuário ou a função, pois ela não corresponde ao ID principal armazenado na política de confiança. Quando isso acontece, o ID principal é exibido no console, pois a AWS não pode mais mapeá-lo de volta para um ARN. O resultado é que, se você excluir e recriar um usuário ou uma função referenciados no elemento `Principal` de uma política de confiança, você deverá editar a função para substituir o nome de recurso da Amazon (ARN). Ele é transformado no novo ID principal quando você salva a política.

Uso de uma política para delegar acesso a serviços

O exemplo a seguir mostra uma política que pode ser anexada a uma função. A política permite que dois serviços, Amazon EMR e AWS Data Pipeline, assumam a função. Os serviços podem então realizar qualquer tarefa concedida pela política de permissões atribuída à função (não exibida). Para especificar vários principais de serviço, você não especifica dois elementos `Service`; pode ter apenas um. Em vez disso, você usa uma variedade de principais de serviços múltiplas como o valor de um único elemento `Service`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
```

```
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Uso de uma política baseada em recurso para delegar acesso a um bucket do Amazon S3 em outra conta

Neste exemplo, a conta A usa uma política baseada em recurso (uma política de bucket do Amazon S3) para conceder à conta B o acesso total ao bucket do S3 da conta A. Em seguida, a conta B cria uma política de usuário do IAM para delegar esse acesso ao bucket da conta A a um dos usuários na conta B.

A política de bucket do S3 na conta A pode se parecer com a seguinte política. Neste exemplo, o bucket do S3 da conta A é denominado amzn-s3-demo-bucket e o número da conta B é 111122223333. Ele não especifica quaisquer usuários individuais ou grupos na conta B, apenas a conta em si.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Como alternativa, a conta A pode usar as Listas de controle de acesso (ACLs) do Amazon S3 para conceder à conta B acesso ao bucket do S3 ou a um único objeto de um bucket. Nesse caso, a única coisa que muda é como a conta A concede acesso à conta B. A conta B ainda usa uma política para delegar acesso a um grupo do IAM na conta B, conforme descrito na próxima parte

deste exemplo. Para mais informações sobre como controlar o acesso em buckets e objetos do S3, consulte Controle de acesso no Guia do usuário do Amazon Simple Storage Service.

O administrador da conta B pode criar a seguinte política de exemplo. A política concede acesso de leitura a um grupo ou usuário na conta B. A política anterior concede acesso à conta B. No entanto, grupos e usuários individuais na conta B não podem acessar o recurso até que uma política de grupo ou usuário explicitamente conceda permissões ao recurso. As permissões nesta política podem ser apenas um subconjunto das permissões anteriores de políticas entre contas. A conta B não pode conceder mais permissões para seus grupos e usuários do que a conta A concedeu para a conta B na primeira política. Nesta política, o elemento `Action` é explicitamente definido para permitir apenas ações `List` e o elemento `Resource` desta política corresponde ao `Resource` para a política de bucket implantada pela conta A.

Para implementar essa política, a conta B usa o IAM para anexá-la ao usuário (ou grupo) apropriado na conta B.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/*"
    ]
  }
}
```

Uso de uma política baseada em recurso para delegar acesso a uma fila do Amazon SQS em outra conta

No exemplo a seguir, a conta A tem uma fila do Amazon SQS que usa uma política baseada em recurso anexada à fila para conceder acesso à fila para a conta B. Em seguida, a conta B usa uma política de grupo do IAM para delegar acesso a um grupo na conta B.

O seguinte exemplo de política de fila concede à conta B permissão para realizar as ações `SendMessage` e `ReceiveMessage` na fila da conta A chamada queue1, mas apenas entre o meio-

dia e 15:00 em 30 de novembro de 2014. O número de conta da conta B é 1111-2222-3333. A conta A usa o Amazon SQS para implementar esta política.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

A política da conta B para delegar acesso a um grupo na conta B pode se parecer com o exemplo a seguir. A conta B usa o IAM para anexar essa política a um grupo (ou usuário).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

No exemplo de política de usuário do IAM anterior, a conta B usa um caractere curinga para conceder a seu usuário acesso a todas as ações do Amazon SQS na fila da conta A. No entanto,

a conta B pode delegar acesso somente na medida em que o acesso foi concedido a ela. O grupo da conta B que tem a segunda política pode acessar a fila apenas entre meio-dia e 15:00 em 30 de novembro de 2014. O usuário só pode executar as ações `SendMessage` e `ReceiveMessage`, conforme definido na política de fila do Amazon SQS da conta A.

Não é possível delegar acesso quando o acesso à conta é negado

Uma Conta da AWS não pode delegar acesso aos recursos de outra conta se a outra conta tiver explicitamente negado o acesso à conta pai do usuário. A negação se propaga para os usuários daquela conta, independentemente de terem ou não políticas existentes que concedam acesso a eles.

Por exemplo, a conta A grava uma política de bucket no bucket do S3 da conta A que explicitamente nega o acesso da conta B ao bucket da conta A. Mas a conta B escreve uma política de usuário do IAM que concede a um usuário na conta B acesso ao bucket da conta A. A negação explícita aplicada ao bucket do S3 da conta A se propaga para os usuários da conta B e substitui a política de usuário do IAM que concede acesso ao usuário da conta B. (Para obter informações detalhadas sobre como as permissões são avaliadas, consulte Lógica da avaliação de política.)

A política de bucket da conta A pode se parecer com a seguinte política. Neste exemplo, o bucket do S3 da conta A é denominado amzn-s3-demo-bucket e o número da conta B é 1111-2222-3333.. A conta A usa o Amazon S3 para implementar esta política.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
  }
}
```

Esta negação explícita substitui todas as políticas na conta B que concedem permissão para acessar o bucket do S3 na conta A.

Criar perfis automaticamente com o gerenciador de perfis

Muitos serviços da AWS dependem de perfis do IAM para realizar ações em seu nome. O gerenciador de perfis do IAM é uma configuração de conta opcional que provisiona esses perfis automaticamente para que você não precise configurá-los. O gerenciador de perfis foi projetado para ajudar você a começar a criar com os serviços da AWS sem a configuração manual adicional do IAM.

O gerenciador de perfis está disponível em alguns consoles de serviço da AWS, listados em the section called “Consoles de serviço da AWS compatíveis com o gerenciador de perfis” nesta página.

Como funciona o gerenciador de perfis

Quando o gerenciador de perfis está habilitado, ele fornece os perfis apropriados ao serviço e ao recurso como parte do fluxo de trabalho relevante do console. Os consoles de serviço compatíveis indicarão quando o gerenciador de perfis estiver disponível. Nenhuma ação adicional é exigida para aceitar o perfil fornecido pelo gerenciador de perfis. Por exemplo, quando você cria uma função do Lambda da AWS no console, o Lambda usa o gerenciador de perfis para criar o perfil de execução da função por padrão. Para usar um perfil diferente, escolha um perfil existente ou crie um novo manualmente.

O gerenciador de perfis funciona aplicando um modelo de perfil, uma definição da política de confiança de um perfil e permissões para um serviço e caso de uso. Para saber mais sobre o modelo que cada serviço usa, consulte Visão geral dos modelos de perfil. Um modelo para uma tarefa definida concede somente as ações de que a tarefa precisa. Um modelo para trabalhos abertos, como executar seu próprio código, concede amplo acesso às APIs da AWS para que você possa criar sem conceder acesso a cada serviço.

Os perfis do IAM criados pelo gerenciador de perfis se comportam como qualquer outro perfil criado por você. Eles podem ser visualizados, editados ou excluídos. Cada perfil também registra o modelo de perfil de onde veio, para que você possa identificar os perfis que o gerenciador de perfis criou. AWS O CloudTrail gera um log de quando o gerenciador de perfis cria perfis; assim, você pode auditar quando e como cada perfil foi criado. A criação de perfil do gerenciamento de perfis aparece no CloudTrail como o evento AcquireRole.

Pré-requisitos

Habilitar o gerenciador de perfis requer as permissões `iam:PutAccountProperties` e `iam:CreateServiceLinkedRole`. Desabilitar o gerenciador de perfis requer a permissão `iam:PutAccountProperties`. Visualizar a configuração atual requer a permissão

`iam:GetAccountProperties`. A política gerenciada `IAMFullAccess` da AWS inclui todas essas permissões. Para todas as permissões exigidas pelo gerenciador de perfis, consulte Gerenciar o acesso ao gerenciador de perfis.

Como habilitar e desabilitar o gerenciador de perfis

Você pode habilitar e desabilitar o gerenciador de perfis usando o Console de gerenciamento da AWS, a AWS CLI ou a API da AWS. O gerenciador de perfis é controlado pela propriedade no nível da conta `RoleManager/Enabled`, que você define usando a API `PutAccountProperties`. Um valor de `true` habilita o gerenciador de perfis; `false` desabilita-o.

> **Note**
>
> Na maioria das contas da AWS, o gerenciador de perfis está desabilitado por padrão. Porém, se você criou sua conta usando nossa nova experiência da AWS, o gerenciador de perfis estará habilitado por padrão e não poderá ser desabilitado até que você ative os atributos avançados. Para obter mais informações, consulte Ativar atributos avançados da AWS.

Para habilitar ou desabilitar o gerenciador de perfis (console)

O gerenciador de perfis pode ser habilitado ou desabilitado na página Configurações da conta no console do IAM. Essa página também mostra se o gerenciador de perfis está habilitado para a conta.

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM.
2. No painel de navegação, selecione Configurações da conta.
3. Na seção gerenciador de perfis, escolha Habilitar ou Desabilitar.

Os perfis criados pelo gerenciador de perfis não são excluídos quando você desabilita o gerenciador de perfis.

Para habilitar ou desabilitar o gerenciador de perfis (AWS CLI)

Defina a propriedade `RoleManager/Enabled` como `true` para habilitar o gerenciador de perfis ou `false` para desativá-lo.

- Para habilitar o gerenciador de perfis, execute `put-account-properties`:

```
aws iam put-account-properties --properties RoleManager/Enabled=true
```

- Para desabiltiar o gerenciador de perfis, execute `put-account-properties`:

```
aws iam put-account-properties --properties RoleManager/Enabled=false
```

- Para ver se o gerenciador de perfis está habilitado para a conta, execute `get-account-properties`:

```
aws iam get-account-properties
```

Para habilitar ou desabilitar o gerenciador de perfis (API da AWS)

- Para habilitar ou desabilitar o gerenciador de perfis, chame PutAccountProperties com a propriedade `RoleManager/Enabled` definida como `true` ou `false`.
- Para ver a configuração atual, chame GetAccountProperties.

Para obter mais informações, consulte PutAccountProperties e GetAccountProperties na Referência de API do IAM.

Consoles de serviço da AWS compatíveis com o gerenciador de perfis

O gerenciador de perfis oferece suporte à criação de perfis para os seguintes consoles de serviço da AWS:

- AWS Backup
- AWS CloudFormation
- Amazon CloudWatch
- Amazon EKS
- AWS Elastic Beanstalk
- Amazon EventBridge
- AWS Glue
- AWS Lambda
- Amazon RDS
- Estúdio Unificado Amazon SageMaker
- AWS Secrets Manager

- AWSFunções de etapa da

A AWS continuará adicionando suporte de serviço ao longo do tempo. Para saber mais sobre o modelo de perfil que cada serviço usa, consulte Visão geral dos modelos de perfil.

Informações relacionadas

- Gerenciar o acesso ao gerenciador de perfis
- Aplicar permissões de privilégio mínimo a um perfil criado automaticamente
- Visão geral dos modelos de perfil

Gerenciar o acesso ao gerenciador de perfis

Você controla o acesso ao gerenciador de perfis em dois níveis. As permissões do IAM determinam quem em uma conta pode habilitar o gerenciador de perfis e quem pode usar os perfis que ele fornece. Em uma organização, as políticas de controle de serviços (SCPs) determinam o que as contas de membros podem fazer.

Permissões para o gerenciador de perfis

Para habilitar ou desabilitar o gerenciador de perfis, você precisa da permissão `iam:PutAccountProperties`. A política gerenciada `IAMFullAccess` da AWS inclui isso.

Para usar o gerenciador de perfis, o usuário precisa das permissões do IAM para as ações que o gerenciador de perfis executa em seu nome. O gerenciador de perfis fornece perfis por meio da API `AcquireRole`, que avalia cada ação subjacente do IAM em relação às permissões do usuário:

- Se um perfil que corresponder ao modelo já existir na conta, `AcquireRole` retorna esse perfil. Isso requer a permissão `iam:GetRole`.
- Se nenhum perfil correspondente existir, `AcquireRole` cria um com base no modelo. Isso requer a permissão `iam:CreateRole`, mais as permissões para o que o modelo define: `iam:PutRolePolicy` se o modelo adicionar políticas embutidas e `iam:AttachRolePolicy` se o modelo anexar políticas gerenciadas.

Um usuário com acesso total ao IAM tem todas as permissões exigidas pelo gerenciador de perfis. Para ver as ações exatas que um modelo específico exige, recupere o modelo com `GetRoleTemplateVersion` e revise as respectivas políticas.

Quando o gerenciador de perfis cria um perfil, o AWS CloudTrail registra a criação como um único evento de `AcquireRole`. O evento mostra quem chamou a operação, o modelo de perfil, os valores de parâmetros usados e o perfil criado.

O exemplo a seguir mostra que o gerenciador de perfis chamou a operação `AcquireRole` em nome de um usuário que assumiu o perfil `Admin`, usando o modelo de perfil `PowerUserRoleTemplate` para criar o perfil `PowerUserRole`.


```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123EXAMPLE:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "accountId": "123456789012",
        "accessKeyId": "ASIAWZYMLEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2026-08-11T18:18:50Z",
                "mfaAuthenticated": "false"
            }
        },
        "onBehalfOf": {
            "userId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "identityStoreArn": "arn:aws:identitystore::111122223333:identitystore/
a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
        }
    },
    "eventTime": "2026-08-11T18:23:00Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "AcquireRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/151.0.0.0 Safari/537.36",
```

```
    "requestParameters": {
        "templateArn": "arn:aws:iam::aws:role-template/iam.amazonaws.com/
PowerUserRoleTemplate:1",
        "templateMinorVersion": 0,
        "replacementValues": {
            "AWSServiceName": {
                "values": [
                    "iam.amazonaws.com"
                ]
            }
        }
    },
    "responseElements": {
        "role": {
            "path": "/",
            "roleName": "PowerUserRole",
            "roleId": "AROA987EXAMPLE",
            "arn": "arn:aws:iam::123456789012:role/PowerUserRole",
            "createDate": "2026-08-11T18:23:00Z",
            "assumeRolePolicyDocument": "%7B%22Version%22:%222012-10-17%22,
%22Statement%22:%5B%7B%22Effect%22:%22Allow%22,%22Principal%22:%7B%22Service%22:
%22iam.amazonaws.com%22%7D,%22Action%22:%22sts:AssumeRole%22%7D%5D%7D"
        }
    },
    "requestID": "fde3dac0-73ac-491c-876b-EXAMPLE89c57",
    "eventID": "c93332cf-1429-4aee-bcaa-EXAMPLE3790d",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```


Controlar o gerenciador de perfis em uma organização

Você pode controlar o gerenciador de perfis em uma organização com uma política de controle de serviços (SCP) da AWS Organizations. Os exemplos a seguir mostram como bloquear a ativação do

gerenciador de perfis, bloquear a criação de perfis ou bloquear somente a criação de perfis com base em modelos.

Esses exemplos usam a estratégia de lista de negação, o que significa que você também precisa de `FullAWSAccess` ou de outra política que permita ações anexadas às entidades da sua organização. Você ainda precisa conceder permissões às suas entidades principais com políticas baseadas em identidade.

Para evitar que as contas de membros habilitem o gerenciador de perfis por meio do console ou da API, negue a permissão `iam:PutAccountProperties`. Recomendamos essa política quando você cria novas contas diretamente por meio do Organizations.

```
{
    "Sid": "DenyRoleManagerEnablement",
    "Effect": "Deny",
    "Action": "iam:PutAccountProperties",
    "Resource": "*",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "iam:AccountPropertyNamespaces": "RoleManager"
        }
    }
}
```

Bloquear o acesso ao uso do gerenciador de perfis

Para bloquear o uso do gerenciador de perfis, use a chave de contexto `iam:RoleTemplateARN`, que está presente nas ações do IAM que criam um perfil com base em um modelo. A política a seguir nega todas as ações do IAM que tenham um valor não nulo para a chave de contexto `iam:RoleTemplateARN`. Essa política não impede toda a criação de perfis. Ela bloqueia somente criações de perfis por meio do gerenciador de perfis e com base em modelos por meio do SDK.

```
{
    "Sid": "DenyTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": "iam:*",
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": false
        }
```

```
    }
}
```

Bloquear a criação de perfis, exceto por meio do gerenciador de perfis

Para permitir a criação de perfis somente por meio do gerenciador de perfis, use a política a seguir para bloquear a criação de perfis que não seja feita pelo gerenciador de perfis. Essa política ainda permite a criação de perfis baseados em modelos diretamente por meio do SDK.

```
{
    "Sid": "DenyNonTemplatedRoleCreation",
    "Effect": "Deny",
    "Action": [
        "iam:CreateRole",
        "iam:AttachRolePolicy",
        "iam:PutRolePolicy",
        "iam:PutRolePermissionsBoundary",
        "iam:TagRole"
    ],
    "Resource": "*",
    "Condition": {
        "Null": {
            "iam:RoleTemplateARN": true
        }
    }
}
```

Informações relacionadas

- Criar perfis automaticamente com o gerenciador de perfis
- Aplicar permissões de privilégio mínimo a um perfil criado automaticamente
- Visão geral dos modelos de perfil

Aplicar permissões de privilégio mínimo a um perfil criado automaticamente

Ao criar um recurso em um console de serviço, o gerenciador de perfis fornece o perfil do IAM de que o recurso precisa. Para algumas tarefas, como executar seu próprio código, a AWS não pode determinar com antecedência quais permissões são exigidas. Como resultado, um perfil pode permitir permissões além daquelas que o recurso exige. Depois de saber de quais permissões o recurso precisa, você pode reduzir as permissões do perfil para que correspondam.

A desativação do gerenciador de perfis não afeta os perfis criados anteriormente pelo gerenciador de perfis. Você pode desativar o gerenciador de perfis sem reduzir as permissões e reduzir as permissões de um perfil sem desativar o gerenciador de perfis.

Quando aplicar permissões de privilégio mínimo

Conceder a um perfil somente as permissões que seu recurso exige é uma prática recomendada de segurança, de acordo com o princípio do privilégio mínimo. Para a maioria dos perfis, o gerenciador de perfis cria perfis com permissões bem definidas. Para alguns perfis, como os de recursos computacionais ou de gerenciamento da infraestrutura de nuvem, ele cria perfis com permissões amplas.

Quando estiver tudo pronto para começar a definir o escopo dos perfis criados pelo gerenciador de perfis, recomendamos começar com os perfis associados às workloads e recursos mais confidenciais, como os que lidam com dados de PII ou de produção. Para saber mais, consulte Preparar para permissões de privilégio mínimo.

Como analisar o acesso não utilizado

Antes de desabilitar o gerenciador de perfis, a AWS recomenda que você use o IAM Access Analyzer para criar um analisador de acesso não usado em sua conta. O analisador compara as ações que um perfil pode realizar com as ações que ele executou e relata as permissões não utilizadas. O analisador analisa os perfis do IAM em sua conta, incluindo aqueles que você criou.

O local em que as descobertas aparecem depende de como o analisador foi fornecido:

- Analisador fornecido pelo gerenciamento de perfis: quando a AWS fornece o analisador de acesso não usado em sua conta (consulte a observação nesta seção), as descobertas aparecem na página Perfis no console do IAM. Para cada perfil, é possível ver o número de permissões não utilizadas e abrir uma recomendação. Use o procedimento a seguir para reduzir as permissões de um perfil na página Perfis. Depois de desabilitar o gerenciador de perfis, a página Configurações da conta fornece um botão Visualizar perfis que abre a página Perfis, onde as descobertas de cada perfil aparecem.
- Analisador criado por você mesmo: quando você cria seu próprio analisador de acesso não usado, as descobertas e recomendações aparecem no console do Analisador de acesso do IAM. Essa experiência não é alterada pelo gerenciamento de perfis. Para analisar e agir com base nas descobertas de acesso não usado, consulte Descobertas do Analisador de acesso do IAM.

> **Note**
>
> Se o AWS habilitou o gerenciador de perfis para sua conta criada usando a nova experiência da AWS, você não precisará criar um analisador na primeira vez em que desativar o gerenciador de perfis. Depois que você ativar os atributos avançados e desativar o gerenciador de perfis, a AWS fornece um analisador de acesso não utilizado sem custo adicional por 90 dias. O analisador oferece visibilidade das permissões de perfil não usadas e das recomendações de redução do escopo de políticas; assim, você pode atualizar os perfis em sua conta, incluindo os que o gerenciador de perfis criou, para obter o privilégio mínimo.

Pré-requisitos

Para revisar as descobertas do Access Analyzer, você precisa de permissão para realizar a ação `access-analyzer:ListFindings`. Para aplicar uma recomendação, você precisa da permissão para executar `iam:CreatePolicy`, `iam:AttachRolePolicy` e `iam:DetachRolePolicy` no perfil.

Para reduzir as permissões de um perfil no console (analisador fornecido pelo gerenciamento de perfis)

Essas etapas se aplicam quando a AWS fornece o analisador de acesso não usado para sua conta. Se você criou seu próprio analisador, revise e aplique as recomendações no console do Analisador de acesso do IAM. Consulte Descobertas do Analisador de acesso do IAM.

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM.
2. No painel de navegação, escolha Perfis.
3. Na página Perfis, na coluna Descobertas, escolha o link Permissões não usadas para o perfil cujas permissões você deseja reduzir.
4. Na página Reduzir permissões de perfil, revise as permissões não usadas relatadas para o perfil.
5. Compare as permissões recomendadas com as permissões atuais do perfil e revise as ações que a recomendação remove.
6. Escolha Fazer alterações.

Como as permissões recomendadas funcionam

Com base nas descobertas do analisador de acesso não utilizado, o IAM cria uma ou mais políticas gerenciadas pelo cliente que contêm as permissões recomendadas. O IAM anexa as novas políticas antes de desanexar as antigas; portanto, o perfil mantém suas permissões durante toda a mudança.

Por exemplo, considere um perfil de execução do Lambda da AWS. O IAM Access Analyzer relata quais permissões do perfil não foram usadas e recomenda uma política com escopo para as ações que o perfil executou, como gravar logs no Amazon CloudWatch Logs. Você analisa a recomendação em relação ao que a função executa, confirma se cobre tudo o que a função precisa e a aplica. Em seguida, o perfil permite somente as ações de que a função precisa.

> **Important**
>
> A recomendação é baseada nos últimos 30 dias de atividade. Uma permissão que o recurso usa com pouca frequência em um período mais longo, como para um trabalho trimestral, pode aparecer como não usada. Analise a recomendação em relação às necessidades da sua workload antes de aplicá-la.
>
> Depois de alterar as políticas do perfil ou políticas de confiança, o gerenciador de perfis não gerencia mais esse perfil. O perfil não incorporará nenhuma alteração de versão ao modelo de perfil originalmente usado para criar o perfil.

O IAM limita o número de políticas gerenciadas que você pode anexar a um perfil. Se a aplicação da recomendação exceder esse limite, o console notificará você e não aplicará a alteração. Para obter informações sobre essa cota, consulte IAM e cotas do AWS STS.

Informações relacionadas

- Criar perfis automaticamente com o gerenciador de perfis
- Gerenciar o acesso ao gerenciador de perfis
- Visão geral dos modelos de perfil

Visão geral dos modelos de perfil

Um modelo de perfil é um esquema de perfil usado para criar novos perfis. O modelo define a política de confiança e as permissões do perfil; assim, você obtém um perfil pronto para uso em um serviço sem precisar escrever as políticas por conta própria.

O gerenciador de perfis usa modelos de perfil gerenciados pela AWS, mas você também pode usar modelos de perfil sem o gerenciador de perfis. Esta página explica como funcionam os modelos de perfil e lista o modelo para cada serviço.

Como funcionam os modelos de perfil

Um modelo de perfil contém uma política de confiança e permissões, além do nome do recurso da Amazon (ARN). A política de confiança define quais entidades principais podem assumir o perfil. As permissões definem o que o perfil pode fazer. As versões do modelo são imutáveis. Quando AWS atualiza um modelo, publica uma nova versão em vez de alterar uma existente.

Os modelos de perfil podem aceitar parâmetros adicionais para personalizar e definir o escopo das propriedades e permissões do perfil. Por exemplo, um modelo pode assumir um `RoleName` para o perfil que ele cria ou um identificador de recurso, como `bucketName`. Quando o gerenciador de perfis cria um perfil no console, ele aplica os parâmetros às permissões e à política de confiança do perfil. Ao criar um perfil com base em um modelo diretamente com a API do IAM, você fornece esses parâmetros.

Para criar um perfil com base em um modelo, use a API `iam:AcquireRole` com o ARN do modelo e transmita os parâmetros que o modelo exige.

As definições de um modelo de perfil podem mudar entre as versões. Para ver a política de confiança atual e as permissões de um modelo, recupere-o com `iam:GetRoleTemplateVersion` usando o ARN do modelo. A AWS recomenda inspecionar o modelo antes de confiar nele, especialmente para um modelo que conceda amplo acesso.

Como o gerenciador de perfis usa modelos

Ao criar um recurso em um console de serviço compatível, o gerenciador de perfis fornece um perfil do modelo que corresponde ao serviço e ao caso de uso. O gerenciador de perfis escolhe o modelo e fornece os parâmetros para você. O resultado é o mesmo perfil que você obteria ao chamar `iam:AcquireRole`. Para habilitar o gerenciador de perfis, consulte Gerenciar o acesso ao gerenciador de perfis.

Diretório de modelos de perfil

A tabela a seguir lista o modelo de perfil para cada serviço ao qual o gerenciador de perfis oferece suporte, com o ARN do modelo e os parâmetros que ele usa. Use o ARN para criar um perfil com base no modelo ou para analisar qual perfil o gerenciador concede para esse serviço. Os serviços são listados em ordem alfabética.

Serviço	Modelo de perfil	ARN do modelo	Parâmetros
AWS Backup	AWSBackupDefaultServiceRoleTemplate	arn:aws:iam::aws:role-template/backup.amazonaws.com/AWSBackupDefaultServiceRoleTemplate:1	accountId, RoleName
AWS CloudFormation	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName, RoleName
Amazon EKS	AmazonEKSAutoClusterRoleTemplate	arn:aws:iam::aws:role-template/eks.amazonaws.com/AmazonEKSAutoClusterRoleTemplate:1	RoleName
Amazon EKS	AmazonEKSAutoNodeRoleTemplate	arn:aws:iam::aws:role-template/ec2.amazonaws.com/AmazonEKSAutoNodeRoleTemplate:1	RoleName
AWS Elastic Beanstalk	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName, RoleName
Amazon EventBridge	PowerUserRoleTemplate	arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1	AWSServiceName, RoleName
AWS Glue	AWSGlueCrawlerS3PathsRoleTemplate	arn:aws:iam::aws:role-template/glue.amazonaws.com/AWSGlueCrawlerS3PathsRoleTemplate:1	RoleName, accountId, s3path
AWS Glue	AWSGluePowerUserRoleTemplate	arn:aws:iam::aws:role-template/glue.amazonaws.com/AWSGluePowerUserRoleTemplate:1	accountId, RoleName

Serviço	Modelo de perfil	ARN do modelo	Parâmetros
AWS Lambda	PowerUser RoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName
Amazon RDS	AmazonRDS DirectoryServiceIntegration RoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSDirectoryServiceIntegrationRoleTemplate:1`	RoleName, accountId
Amazon RDS	AmazonRDS ExportToS3RoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSExportToS3RoleTemplate:1`	RoleName, accountId, bucketName
Amazon RDS	AmazonRDS ImportFromS3RoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSImportFromS3RoleTemplate:1`	CUSTOM_KMS_KEY, RoleName, accountId, bucketName, keyAccountId, keyRegion, kmsKeyId, prefix
Amazon RDS	AmazonRDS MonitoringRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSMonitoringRoleTemplate:1`	RoleName, accountId

Serviço	Modelo de perfil	ARN do modelo	Parâmetros
Amazon RDS	AmazonRDSProxyRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSProxyRoleTemplate:2`	PROXY_WITH_CUSTOM_KEY, PROXY_WITH_DB_CONNECTION, PROXY_WITH_SECRET, RoleName, accountId, dbResourceId, dbUser, keyId, region, secretId
Amazon RDS	AmazonRDSSecretAccessForDMSRoleTemplate	`arn:aws:iam::aws:role-template/rds.amazonaws.com/AmazonRDSSecretAccessForDMSRoleTemplate:1`	CUSTOM_KMS_KEY, RoleName, accountId, kmsKeyId, region, secretId
Estúdio Unificado Amazon SageMaker	AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate	`arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerAdminIAMPermissiveExecutionRoleTemplate:1`	CMK_ENABLED, RoleName, accountId, keyAccountId, keyRegion, kmsKeyId
Estúdio Unificado Amazon SageMaker	AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate	`arn:aws:iam::aws:role-template/datazone.amazonaws.com/AmazonSageMakerUserIAMPermissiveExecutionRoleTemplate:1`	RoleName, accountId

Serviço	Modelo de perfil	ARN do modelo	Parâmetros
AWS Secrets Manager	AWSSecretsManagerRotationRoleTemplate	`arn:aws:iam::aws:role-template/secretsmanager.amazonaws.com/AWSSecretsManagerRotationRoleTemplate:1`	ADMIN_RESOURCE_ENABLED, CMK_ENABLED, RoleName, accountId, adminType, kmsKeyArn, region, resourceType
Amazon CloudWatch	PowerUserRoleTemplate	`arn:aws:iam::aws:role-template/iam.amazonaws.com/PowerUserRoleTemplate:1`	AWSServiceName, RoleName

Informações relacionadas

- Criar perfis automaticamente com o gerenciador de perfis
- Gerenciar o acesso ao gerenciador de perfis
- Referência da API do IAM

Gerenciamento de perfis do IAM

Para que um usuário, uma aplicação ou um serviço possa usar um perfil que você criou, você deverá conceder permissões para alternar para esse perfil. É possível usar qualquer política anexada a grupos ou usuários para conceder as permissões necessárias. Esta seção descreve como conceder aos usuários permissão para usar uma função. Ela também explica como o usuário pode alternar para uma função no Console de gerenciamento da AWS, no Tools for Windows PowerShell, no AWS Command Line Interface (AWS CLI) e na API `AssumeRole`.

> **Important**
>
> Ao criar uma função de forma programática, em vez de no console do IAM, você tem a opção de adicionar um `Path` de até 512 caracteres além do `RoleName`, que pode ter até

64 caracteres. No entanto, se você pretende usar uma função com o recurso Switch Role (Alternar função) no Console de gerenciamento da AWS, o `Path` e o `RoleName` combinados não podem exceder 64 caracteres.

Tópicos

- Visualizar acesso à função
- Gerar uma política com base em informações de acesso
- Conceder permissões a um usuário para alternar perfis
- Conceda permissões a um usuário para passar um perfil para um serviço da AWS
- Revogar as credenciais de segurança temporárias do perfil do IAM
- Atualizar um perfil vinculado ao serviço
- Atualizar a política de confiança de um perfil
- Atualizar permissões para um perfil
- Atualizar as configurações de um perfil
- Excluir perfis ou perfis de instância

Visualizar acesso à função

Antes de alterar as permissões de uma função, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Gerar uma política com base em informações de acesso

Às vezes, você pode conceder permissões a uma entidade do IAM (usuário ou função) além do que é exigido. Para ajudar você a refinar as permissões concedidas, você pode gerar uma política do IAM baseada na atividade de acesso para uma entidade. O IAM Access Analyzer revisa seus logs do AWS CloudTrail e gera um modelo de política que contém as permissões que foram usadas pela entidade no intervalo de datas especificado. Você pode usar o modelo para criar uma política gerenciada com permissões refinadas e anexá-la à entidade do IAM. Dessa forma, você concede apenas as permissões de que o usuário ou a função precisa para interagir com os recursos da AWS

para seu caso de uso específico. Para saber mais, consulte Geração de políticas do IAM Access Analyzer.

Conceder permissões a um usuário para alternar perfis

Ao criar um perfil para acesso entre contas, o administrador estabelece confiança entre a conta que possui perfil, os recursos (conta de confiança) e a conta que contém os usuários (conta confiável). Para fazer isso, o administrador da conta confiável especifica o número da conta de confiança como `Principal` na política de confiança da função. Isso potencialmente permite que qualquer usuário na conta confiável assuma o perfil. Para concluir a configuração, o administrador da conta confiável deve conceder a grupos ou usuários específicos nessa conta permissão para alternar para a função.

Para conceder permissão para alternar para um perfil

1. Como administrador da conta confiável, crie uma nova política para o usuário ou edite uma política existente adicionando os elementos necessários. Para obter detalhes, consulte Criação ou edição da política.
2. Em seguida, escolha como deseja compartilhar as informações do perfil:

   - Link do perfil: envie aos usuários um link que os leve para a página Switch Role (Alternar perfil) com todos os detalhes já preenchidos.
   - Account ID or alias (ID ou alias da conta): forneça a cada usuário o nome do perfil com o número de ID da conta ou o alias da conta. Em seguida, o usuário acessa a página Alternar função e adiciona os detalhes manualmente.

   Para obter detalhes, consulte Fornecer informações ao usuário.

Observe que você só pode alternar funções quando fizer login como um usuário do IAM, uma função federada SAML ou uma função federada de identidade da web. Você não pode mudar de funções quando se conecta como o Usuário raiz da conta da AWS.

> **Important**
>
> Não é possível alternar perfis no Console de gerenciamento da AWS para um perfil que exija um valor ExternalId. Você só pode alternar para tal perfil chamando a API `AssumeRole`, que é compatível com o parâmetro `ExternalId`.

Observações

- Este tópico aborda políticas para um usuário, porque, em última análise, você está concedendo permissões a um usuário para realizar uma tarefa. Porém, não recomendamos conceder permissões diretamente a um usuário individual. Ao assumir um perfil, o usuário recebe as permissões associadas a esse perfil.
- Quando você muda de funções no Console de gerenciamento da AWS, o console sempre usa suas credenciais originais para autorizar a mudança. Isso se aplica se você fizer login como usuário do IAM, como função federada SAML ou como função federada de identidade da Web. Por exemplo, se você mudar para RoleA, o IAM utilizará o usuário original ou as credenciais de perfil federado para determinar se você tem permissão para assumir o RoleA. Se você tentar mudar para a FunçãoB enquanto estiver usando a FunçãoA, o usuário original ou as credenciais de função federada serão usadas para autorizar sua tentativa. As credenciais para RoleA não são usadas para essa ação.

Tópicos

- Criação ou edição da política
- Fornecer informações ao usuário

Criação ou edição da política

Uma política que conceda a um usuário permissão para assumir uma função deve incluir uma instrução com o efeito `Allow` sobre o seguinte:

- A ação `sts:AssumeRole`
- O Amazon Resource Name (ARN – Nome de recurso da Amazon) da função em um elemento `Resource`

Os usuários que obtêm a política têm permissão para alternar perfis no recurso listado (por meio da associação do grupo ou anexado diretamente).

Note

Se `Resource` for definido como `*`, o usuário poderá assumir qualquer função em qualquer conta que confie na conta do usuário. (Em outras palavras, a política de confiança da função

especifica a conta do usuário como `Principal`). Como melhor prática, recomendamos que você siga o princípio do menor privilégio e especifique o ARN completo apenas para as funções de que o usuário precisa.

O exemplo a seguir mostra uma política que permite que o usuário assuma funções em apenas uma conta. Além disso, a política usa um curinga (*) para especificar que o usuário só pode alternar para uma função caso o nome da função comece com as letras `Test`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/Test*"
    }
}
```

Note

As permissões que a função concede ao usuário não são adicionadas às permissões já concedidas ao usuário. Quando um usuário alterna para uma função, ele desiste temporariamente de suas permissões originais em troca das concedidas pela função. Quando o usuário sai da função, as permissões originais do usuário são restauradas automaticamente. Por exemplo, digamos que as permissões do usuário permitam trabalhar com instâncias do Amazon EC2, mas a política de permissões da função não conceda essas permissões. Nesse caso, enquanto usa a função, o usuário não pode trabalhar com instâncias do Amazon EC2 no console. Além disso, as credenciais temporárias obtidas por meio de `AssumeRole` não funcionam com instâncias do Amazon EC2 de maneira programática.

Fornecer informações ao usuário

Depois de criar uma função e conceder ao usuário permissões a fim de alternar para ela, você deverá fornecer ao usuário o seguinte:

- O nome da função
- O ID ou o alias da conta que contém a função

Você pode simplificar o acesso para seus usuários enviando a eles um link pré-configurado com o ID da conta e o nome do perfil. Você pode ver o link do perfil depois de concluir o assistente de Criar perfil selecionando o banner Exibir perfil ou na página Resumo do perfil para qualquer perfil habilitado para várias contas.

Você também pode usar o seguinte formato para construir manualmente o link. Substitua o ID ou o alias da conta e o nome da função para os dois parâmetros no exemplo a seguir.

`https://signin.aws.amazon.com/switchrole?account=`*`your_account_ID_or_alias`*`&roleName=`*`optional_path/role_name`*

Recomendamos direcionar seus usuários a Mudar de um usuário para um perfil do IAM (console) para orientá-los durante o processo. Para solucionar problemas comuns que você pode encontrar ao assumir um perfil, consulte Não consigo assumir uma função.

Considerações

- Se você criar o perfil de forma programática, poderá criá-lo com um caminho e um nome. Se você fizer isso, deverá fornecer o caminho completo e o nome da função aos usuários para que eles possam inseri-los na página Switch Role (Alternar função) do Console de gerenciamento da AWS. Por exemplo: `division_abc/subdivision_efg/role_XYZ`.
- Se você criar um perfil de forma programática, poderá adicionar um `Path` de até 512 caracteres e um `RoleName`. O nome da função pode ter até 64 caracteres. No entanto, para usar uma função com o recurso Switch Role (Alternar função) no Console de gerenciamento da AWS, o `Path` e o `RoleName` combinados não podem exceder 64 caracteres.

- Por motivos de segurança, você pode revisar logs do AWS CloudTrail para saber quem executou uma ação na AWS. Você pode usar a chave de condição `sts:SourceIdentity` na política de confiança da função para exigir que os usuários especifiquem uma identidade quando assumirem uma função. Por exemplo, você pode exigir que os usuários do IAM especifiquem

seu próprio nome de usuário como a identidade-fonte. Isso pode ajudar você a determinar qual usuário executou uma ação específica na AWS. Para obter mais informações, consulte `sts:SourceIdentity`. Você também pode usar `sts:RoleSessionName` para exigir que os usuários especifiquem um nome de sessão quando assumirem uma função. Isso pode ajudar você a diferenciar as sessões de função quando uma função é usada por diferentes entidades de segurança.

Conceda permissões a um usuário para passar um perfil para um serviço da AWS

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Isso permite que o serviço assuma a função posteriormente e realize ações em seu nome. Para a maioria dos serviços, você só precisa passar a função para o serviço uma vez durante a configuração, não toda vez que o serviço assumir a função. Por exemplo, suponha que você tenha uma aplicação em execução em uma instância do Amazon EC2. Esse aplicativo requer credenciais temporárias para autenticação, além de permissões para autorizar o aplicativo a executar ações na AWS. Ao configurar a aplicação, você deve passar uma função para o Amazon EC2 usar com a instância que fornece essas credenciais. Defina as permissões para as aplicações em execução na instância anexando uma política do IAM à função. O aplicativo assume a função sempre que necessário para executar as ações que são permitidas pela função.

Para transmitir uma função (e suas permissões) para um serviço da AWS, um usuário deve ter permissões para transmitir a função para o serviço. Isso ajuda os administradores a garantir que apenas usuários aprovados possam configurar um serviço com uma função que concede permissões. Para permitir que um usuário passe uma função para um produto da AWS, você deve conceder a permissão `PassRole` ao usuário, à função ou ao grupo do IAM do usuário.

> **Warning**
>
> - A permissão `PassRole` só pode ser usada para passar um perfil do IAM para um serviço que compartilha a mesma conta da AWS. Para passar um perfil na Conta A para um serviço na Conta B, primeiro é necessário criar um perfil do IAM na Conta B que possa assumir o perfil da Conta A. Em seguida, o perfil na Conta B pode ser passado para o serviço. Para obter detalhes, consulte Acesso a recursos entre contas no IAM.
> - Não tente controlar quem pode passar por uma função marcando a função e, em seguida, usando a chave de condição `ResourceTag` em uma política com a ação `iam:PassRole`. Os resultados dessa abordagem não são confiáveis.

Ao definir a permissão `PassRole`, é necessário garantir que um usuário não passe um perfil em que o perfil tenha mais permissões do que você deseja que o usuário tenha. Por exemplo, Alice pode não ter permissão para realizar nenhuma ação do Amazon S3. Se Alice pudesse passar um perfil para um serviço que permite ações do Amazon S3, o serviço poderia realizar ações do Amazon S3 em nome de Alice ao executar o trabalho.

Ao especificar um perfil vinculado ao serviço, você também precisa ter permissão para atribuir o perfil ao serviço. Alguns serviços criam automaticamente uma função vinculada ao serviço na sua conta quando você executa uma ação nesse serviço. Por exemplo, o Amazon EC2 Auto Scaling cria a função vinculada ao serviço `AWSServiceRoleForAutoScaling` quando você cria um grupo do Auto Scaling pela primeira vez. Se tentar especificar o perfil vinculado ao serviço ao criar um grupo do Auto Scaling e não tiver a permissão `iam:PassRole`, você receberá um erro. Se você não especificar explicitamente o perfil, a permissão `iam:PassRole` não será necessária, e o padrão é usar o perfil `AWSServiceRoleForAutoScaling` para todas as operações executadas no grupo. Para saber quais serviços dão suporte a funções vinculadas ao serviço, consulte AWSServiços da que funcionam com o IAM. Para saber quais serviços criam automaticamente uma função vinculada quando você executa uma ação no serviço, escolha o link Sim e visualize a documentação das funções vinculadas a serviços para o serviço.

Um usuário pode transmitir um ARN da função como um parâmetro em qualquer operação da API que usa a função para atribuir permissões ao serviço. Em seguida, o serviço verifica se esse usuário tem a permissão `iam:PassRole`. Para limitar o usuário a passar apenas as funções aprovadas, filtre a permissão `iam:PassRole` com o elemento `Resources` da instrução de política do IAM.

Você pode usar o elemento `Condition` em uma política JSON para testar o valor das chaves incluídas no contexto de solicitação de todas as solicitações da AWS. Para saber mais sobre como usar chaves de condição em uma política, consulte Elementos de política JSON do IAM: Condition. A chave de condição `iam:PassedToService` pode ser usada para especificar o principal de serviço do serviço para o qual uma função pode ser passada. Para saber mais sobre como usar a chave de `iam:PassedToService` condição em uma política, consulte iam:PassedToService.

Exemplo 1

Suponha que você deseja conceder a um usuário a capacidade de transmitir qualquer função de um conjunto de funções aprovadas para o serviço Amazon EC2 ao executar uma instância. Três elementos são necessários:

- Uma política de permissões do IAM anexada à função que determina o que a função pode fazer. Defina as permissões para apenas as ações que a função deve realizar e os recursos que a

função precisa para essas ações. Você pode usar a política de permissões do IAM gerenciada pela AWS ou criada para o cliente.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [ "A list of the permissions the role is allowed to use" ],
        "Resource": [ "A list of the resources the role is allowed to access" ]
    }
}
```

- Uma política de confiança para a função que permite que o serviço assuma a função. Por exemplo, você pode anexar a seguinte política de confiança à função com a ação `UpdateAssumeRolePolicy`. Essa política de confiança permite que o Amazon EC2 use a função e as permissões anexadas à função.

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid":
 "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
        "Effect": "Allow",
        "Principal": { "Service": "ec2.amazonaws.com" },
       "Action": "sts:AssumeRole"
    }
}
```

- Uma política de permissões do IAM anexada ao usuário do IAM que permite que o usuário transmita apenas as funções aprovadas. Geralmente `iam:GetRole` é adicionada a `iam:PassRole` para que o usuário possa obter os detalhes da função a ser transmitida. Neste exemplo, o usuário pode transmitir apenas funções que existam na conta especificada com nomes começando com `EC2-roles-for-XYZ-`:

  JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/EC2-roles-for-XYZ-*"
        }
    ]
}
```

Agora o usuário pode iniciar uma instância do Amazon EC2 com uma função atribuída. Os aplicativos em execução na instância podem acessar credenciais temporárias para a função por meio de metadados do perfil da instância. As políticas de permissões anexadas à função determinam o que a instância pode fazer.

Exemplo 2

O Amazon Relational Database Service (Amazon RDS) é compatível com um recurso chamado Monitoramento aprimorado. Este recurso permite que o Amazon RDS monitore uma instância de banco de dados usando um agente. Ele também permite que o Amazon RDS registre métricas de log no Amazon CloudWatch Logs. Para habilitar esse recurso, você deve criar uma função de serviço para conceder ao Amazon RDS permissões para monitorar e gravar métricas em seus logs.

Para criar uma função para o monitoramento aprimorado do Amazon RDS

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Selecione Funções e, depois, Criar função.
3. Escolha o tipo de perfil de Serviço da AWS e, em Casos de uso para outros Serviços da AWS, escolha o serviço RDS. Escolha RDS – Enhanced Monitoring (RDS: monitoramento aprimorado) e, em seguida Next (Avançar).
4. Escolha a política de permissões AmazonRDSEnhancedMonitoringRole.
5. Escolha Próximo.
6. Em Role name (Nome da função), insira um nome de função que ajude a identificar a finalidade da função. Os nomes de função devem ser exclusivos em sua Conta da AWS. Ao ser usado em uma política ou como parte de um ARN, o nome de perfil diferencia maiúsculas de minúsculas.

Quando exibida para os clientes no console, por exemplo, como durante o processo de login, o nome de função não diferencia maiúsculas de minúsculas. Como várias entidades podem fazer referência à função, não é possível editar o nome da função depois de criada.

7. (Opcional) Em Description (Descrição), insira uma descrição para o nova função.
8. (Opcional) Adicione metadados ao usuário anexando tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
9. Revise o perfil e escolha Criar perfil.

A função obtém, automaticamente, uma política de confiança que concede as permissões de serviço `monitoring.rds.amazonaws.com` para assumir a função. Depois disso, o Amazon RDS pode executar todas as ações que a política `AmazonRDSEnhancedMonitoringRole` permite.

O usuário que você deseja que acesse o monitoramento aprimorado precisa de uma política que inclua uma declaração que permita que o usuário liste os perfis do RDS e uma declaração que permita ao usuário transmitir a função, como apresentado a seguir. Use o número da sua conta e substitua o nome do perfil pelo nome fornecido na etapa 6.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
    "Effect": "Allow",
    "Action": [ "iam:PassRole" ],
    "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
}
```

Você pode combinar esta instrução com instruções em outra política ou colocá-la em sua própria política. Em vez de especificar que o usuário pode passar qualquer função que comece com `RDS-`, é possível substituir o nome da função no ARN do recurso por um curinga, por exemplo:

```
    "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

Ações **`iam:PassRole`** em logs do AWS CloudTrail

`PassRole` não é uma chamada de API. `PassRole` é uma permissão, o que significa que nenhum log do CloudTrail é gerado para o `PassRole` do IAM. Para analisar quais perfis são transferidos para quais Serviços da AWS no CloudTrail, é necessário analisar o log do CloudTrail que criou ou modificou o recurso da AWS que recebeu o perfil. Por exemplo, um perfil é transferido para uma função do AWS Lambda ao ser criado. O log da ação `CreateFunction` mostra um registro do perfil que foi transferido para a função.

Revogar as credenciais de segurança temporárias do perfil do IAM

> **Warning**
>
> Se você seguir as etapas nesta página, todos os usuários com sessões atuais criadas ao assumiram a função terão o acesso negado a todas as ações e recursos da AWS. Como resultado, os usuários poderão perder trabalho não salvo.

Ao permitir que os usuários acessem o Console de gerenciamento da AWS com um tempo de duração de sessão longo (como 12 horas), suas respectivas credenciais temporárias não expiram com tanta rapidez. Se os usuários expuserem inadvertidamente suas credenciais a um terceiro não autorizado, este terá acesso durante toda a sessão. No entanto, se necessário, você poderá revogar imediatamente todas as permissões para as credenciais da função emitidas antes de um determinado momento. Todas as credenciais temporárias para essa função emitidas antes do tempo especificado se tornam inválidas. Isso força todos os usuários a refazerem a autenticação e solicitar novas credenciais.

> **Note**
>
> Você não pode revogar a sessão para uma função vinculada a serviço.

Quando você revoga permissões para uma função usando o procedimento neste tópico, a AWS anexa uma nova política em linha à função que nega todas as permissões para todas as ações. Ele incluirá uma condição aplicável às restrições somente se o usuário tiver assumido a função antes do momento em que você revogar as permissões. Se o usuário assumir a função depois que você revogar as permissões, a política de negação não se aplicará a esse usuário.

Para obter mais informações sobre acesso negado, consulte Desabilitar permissões de credenciais de segurança temporárias.

> **Important**
>
> Essa política de negação se aplica a todos os usuários da função especificada, não apenas às sessões do console com maior duração.

Permissões mínimas para revogar as permissões de sessão de uma função

Para revogar permissões de sessão de uma função com êxito, você deve ter a permissão `PutRolePolicy` para a função. Isso permite que você anexe a política em linha `AWSRevokeOlderSessions` à função.

Revogar permissões de uma sessão

Você pode revogar as permissões de sessão de um perfil para negar todas as permissões de qualquer usuário que tenha assumido esse perfil.

> **Note**
>
> Não é possível editar perfis no IAM que foram criados a partir de conjuntos de permissões do Centro de Identidade do IAM. É necessário revogar a sessão ativa do conjunto de permissões de um usuário no Centro de Identidade do IAM. Para obter mais informações, consulte Revogar sessões ativas de perfil do IAM criadas por conjuntos de permissões, no Guia do usuário do Centro de Identidade do IAM.

Para negar imediatamente todas as permissões para qualquer usuário atual de credenciais de função

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis e selecione o nome (não a caixa de seleção) da função cujas permissões você deseja revogar.
3. Na página Resumo para a função selecionada, escolha a guia Revogar sessões.
4. Na guia Revogar sessões, selecione Revogar sessões ativas.

5. AWSA pede que você confirme a ação. Marque a caixa de seleção Confirmo que estou revogando todas as sessões ativas para essa função e escolha Revogar sessões ativas.

   O IAM então anexa uma política chamada `AWSRevokeOlderSessions` ao perfil. Depois de escolher Revogar sessões ativas, a política nega todo o acesso aos usuários que assumiram o perfil no passado, bem como em aproximadamente 30 segundos no futuro. Essa escolha de horário futuro leva em consideração o atraso de propagação da política para lidar com uma nova sessão que foi adquirida ou renovada antes que a política atualizada entrasse em vigor em uma determinada região. Os usuários que assumirem o perfil em mais de aproximadamente 30 segundos após você escolher a opção Revogar sessões ativas não serão afetados. Para saber por que as mudanças nem sempre são imediatamente visíveis, consulte As alterações que eu faço nem sempre ficam imediatamente visíveis.

> **Note**
>
> Se, posteriormente, você escolher Revogar sessões ativas novamente, a marca de data e hora da política será atualizada e ela voltará a negar todas as permissões a todos os usuários que assumiram o perfil antes da nova hora especificada.

Os usuários válidos cujas sessões são revogadas dessa forma devem adquirir credenciais temporárias para uma nova sessão para continuar a trabalhar. A AWS CLI armazena em cache as credenciais até que elas expirem. Para forçar a CLI a excluir e atualizar credenciais de cache que não são mais válidas, execute um dos seguintes comandos:

Linux, macOS ou Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Revogar as permissões da sessão antes de uma hora especificada

Também é possível revogar as permissões da sessão a qualquer momento usando a AWS CLI ou o SDK para especificar um valor para a chave `aws:TokenIssueTime` no elemento Condição de uma política.

Essa política nega todas as permissões quando o valor de `aws:TokenIssueTime` é anterior à data e hora especificadas. O valor do `aws:TokenIssueTime` corresponde ao tempo exato em que as credenciais de segurança temporárias foram criadas. O valor `aws:TokenIssueTime` está presente apenas no contexto de solicitações da AWS assinadas com credenciais de segurança temporárias. Portanto, a instrução Negar na política não afeta as solicitações assinadas com as credenciais de longo prazo do usuário do IAM.

Essa política também pode ser anexada a um perfil. Neste caso, a política afeta somente as credenciais de segurança temporárias que foram criadas pela função antes da data e hora especificadas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Os usuários válidos cujas sessões são revogadas dessa forma devem adquirir credenciais temporárias para uma nova sessão para continuar a trabalhar. A AWS CLI armazena em cache as credenciais até que elas expirem. Para forçar a CLI a excluir e atualizar credenciais de cache que não são mais válidas, execute um dos seguintes comandos:

Linux, macOS ou Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Atualizar um perfil vinculado ao serviço

O método que você usa para editar uma função vinculada ao serviço depende do serviço. Alguns serviços podem permitir que você edite as permissões para uma função vinculada ao serviço no console, API ou CLI do serviço. Contudo, depois que você cria uma função vinculada ao serviço, você não pode mudar o nome da função porque várias entidades podem fazer referência à função. Você pode editar a descrição de qualquer função do console, da API ou da CLI do IAM.

Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Para saber se o serviço oferece suporte a edição da função vinculada ao serviço, escolha o link Sim para visualizar a documentação da função vinculada desse serviço.

Editar a descrição de uma função vinculada ao serviço (console)

Você pode usar o console do IAM para editar a descrição de uma função vinculada ao serviço.

Para editar a descrição de uma função vinculada ao serviço (console)

1. No painel de navegação do console do IAM, escolha Perfis.
2. Escolha o nome da função a ser modificada.
3. No extremo direito da Descrição da função, escolha Editar.
4. Insira uma nova descrição na caixa e escolha Salvar.

Editar a descrição de uma função vinculada ao serviço (AWS CLI)

Você pode usar comandos do IAM na AWS CLI para editar a descrição de uma função vinculada ao serviço.

Para alterar a descrição de uma função vinculada ao serviço (AWS CLI)

1. (Opcional) Para visualizar a descrição atual de uma função, execute os comandos a seguir:

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Use o nome da função, não o nome de recurso da Amazon (ARN), para fazer referência às funções com os comandos da CLI. Por exemplo, se uma função tiver o seguinte nome de recurso da Amazon (ARN): `arn:aws:iam::123456789012:role/myrole`, você fará referência à função como **`myrole`**.

2. Para atualizar a descrição de uma função vinculada ao serviço, execute um dos seguintes comandos:

```
aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
```

Editar a descrição de uma função vinculada ao serviço (API da AWS)

Você pode usar a API do AWS para editar a descrição de uma função vinculada ao serviço.

Para alterar a descrição de uma função (AWS API)

1. (Opcional) Para visualizar a descrição atual de a uma função, chame a seguinte operação e especifique o nome da função:

   API do AWS: GetRole
2. Para atualizar a descrição de uma função, chame a seguinte operação e especifique o nome (e a descrição opcional) da função:

   API da AWS: UpdateRole

Atualizar a política de confiança de um perfil

Para alterar quem pode assumir uma função, você deve modificar a política de confiança da função. Você não pode modificar a política de confiança para uma função vinculada a serviço.

> **Observações**
>
> - Se um usuário for listado como principal em uma política de confiança da função, mas não puder assumir a função, verifique o limite de permissões do usuário. Se um limite de permissões for definido para o usuário, ele deverá permitir a ação `sts:AssumeRole`.
> - Para permitir que os usuários assumam novamente o perfil atual em uma sessão de perfil, especifique o ARN do perfil ou o ARN da Conta da AWS como entidade principal na política de confiança do perfil. Os Serviços da AWS que fornecem recursos computacionais, como o Amazon EC2, Amazon ECS, Amazon EKS e Lambda, fornecem credenciais temporárias e atualizam automaticamente essas credenciais. Isso garante que você tenha sempre um conjunto de credenciais válido. Nesses serviços, não é necessário assumir novamente a função atual para obter credenciais temporárias. Porém,

se pretender passar tags de sessão ou uma política de sessão, você precisará assumir novamente a função atual.

Atualizar a política de confiança de um perfil (console)

Para alterar a política de confiança de um perfil no Console de gerenciamento da AWS

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis.
3. Na lista de funções em sua conta, escolha o nome da função que deseja modificar.
4. Escolha a guia Relacionamentos de confiança e, em seguida, escolha Editar política de confiança.
5. Edite a política de confiança, conforme necessário. Para adicionar outras entidades principais que podem assumir a função, especifique-as no elemento `Principal`. Por exemplo, o fragmento de política a seguir mostra como fazer referência a duas Contas da AWS no elemento `Principal`:

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:root",
    "arn:aws:iam::444455556666:root"
  ]
},
```

Se você especificar um principal em outra conta, adicionar uma conta à política de confiança de uma função é apenas metade da tarefa de estabelecer o relacionamento de confiança entre contas. Por padrão, nenhum usuário nas contas confiáveis pode assumir a função. O administrador da conta confiável recém-criada deve conceder aos usuários a permissão para assumir a função. Para fazer isso, o administrador deve criar ou editar uma política que está anexada ao usuário para permitir acesso ao usuário à ação `sts:AssumeRole`. Para obter mais informações, consulte o procedimento a seguir ou Conceder permissões a um usuário para alternar perfis.

O trecho da política a seguir mostra como referenciar dois produtos da AWS no elemento `Principal`:

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. Ao concluir a edição da política de confiança, escolha Update policy(Atualizar política) para salvar as alterações.

   Para obter mais informações sobre a estrutura e a sintaxe da política, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Para permitir que os usuários em uma conta externa confiável usem a função (console)

Para obter mais informações e detalhes sobre esse procedimento, consulte Conceder permissões a um usuário para alternar perfis.

1. Faz login na externa confiável Conta da AWS.
2. Decida se deseja anexar as permissões a um usuário ou a um grupo. No painel de navegação do console do IAM, escolha Users (Usuários) ou Groups (Grupos) conforme o caso.
3. Escolha o nome do usuário ou do grupo ao qual você deseja conceder acesso e, em seguida, selecione a guia Permissões.
4. Execute um destes procedimentos:

   - Para editar uma política gerenciada pelo cliente, escolha o nome da política, escolha Editar política e, em seguida, selecione a guia JSON. Você não pode editar uma política AWS gerenciada. As políticas AWS gerenciadas são exibidas com o ícone da AWS ( ).
     Para obter mais informações sobre a diferença entre políticas gerenciadas pela AWS e pelo cliente, consulte Políticas gerenciadas e em linha.
   - Para editar uma política em linha, escolha a seta próxima ao nome da política e escolha Editar política.

5. No editor de políticas, adicione um novo elemento `Statement` que especifica o seguinte:

```
{
  "Effect": "Allow",
  "Action": "sts:AssumeRole",
  "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
}
```

   Substitua o ARN na instrução pelo ARN da função que o usuário pode assumir.
6. Siga os prompts na tela para terminar de editar a política.

Atualizar a política de confiança de um perfil (AWS CLI)

Você pode usar a AWS CLI para alterar quem pode assumir um perfil.

Como modificar uma política de confiança da função (AWS CLI)

1. (Opcional) Se você não souber o nome da função que deseja modificar, execute o seguinte comando para listar as funções em sua conta:

   - aws iam list-roles
2. (Opcional) Para visualizar a política de confiança atual de uma função, execute o seguinte comando:

   - aws iam get-role
3. Para modificar as entidades principais confiáveis que podem acessar a função, crie um arquivo de texto com a política de confiança atualizada. É possível usar qualquer editor de texto para construir a política.

   Por exemplo, a seguinte política de confiança mostra como fazer referência a duas Contas da AWS no elemento `Principal`. Isso permite que os usuários de duas Contas da AWS separadas assumam esse perfil.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
```

```
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

Se você especificar um principal em outra conta, adicionar uma conta à política de confiança de uma função é apenas metade da tarefa de estabelecer o relacionamento de confiança entre contas. Por padrão, nenhum usuário nas contas confiáveis pode assumir a função. O administrador da conta confiável recém-criada deve conceder aos usuários a permissão para assumir a função. Para fazer isso, o administrador deve criar ou editar uma política que está anexada ao usuário para permitir acesso ao usuário à ação `sts:AssumeRole`. Para obter mais informações, consulte o procedimento a seguir ou Conceder permissões a um usuário para alternar perfis.

4. Para usar o arquivo que você acabou de criar para atualizar a política de confiança, execute o seguinte comando:

   - aws iam update-assume-role-policy

Para permitir que os usuários em uma conta externa confiável usem a função (AWS CLI)

Para obter mais informações e detalhes sobre esse procedimento, consulte Conceder permissões a um usuário para alternar perfis.

1. Crie um arquivo JSON que contenha uma política de permissões que concede permissões para assumir a função. Por exemplo, a seguinte política contém as permissões necessárias mínimas:

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::111122223333:role/ROLE-NAME"
    }
```

```
}
```

Substitua o ARN na instrução pelo ARN da função que o usuário pode assumir.

2. Execute o seguinte comando para carregar o arquivo JSON que contém a política de confiança para o IAM:

   - aws iam create-policy

   O resultado desse comando inclui o ARN da política. Anote esse ARN, pois você precisará dele em uma etapa posterior.

3. Decida qual usuário ou grupo ao qual anexar a política. Se você não souber o nome do usuário ou do grupo pretendido, use um dos seguintes comandos para listar os usuários ou os grupos em sua conta:

   - aws iam list-users
   - aws iam list-groups

4. Use um dos seguintes comandos para anexar a política criada na etapa anterior ao usuário ou ao grupo:

   - aws iam attach-user-policy
   - aws iam attach-group-policy

Atualizar a política de confiança de um perfil (API da AWS)

Você pode usar a API da AWS para alterar quem pode assumir um perfil.

Como modificar a política de confiança de uma função (API da AWS)

1. (Opcional) Se você não souber o nome da função que deseja modificar, chame a seguinte operação para listar as funções em sua conta:

   - ListRoles

2. (Opcional) Para visualizar a política de confiança atual de uma função, chame a seguinte operação:

   - GetRole

3. Para modificar as entidades principais confiáveis que podem acessar a função, crie um arquivo de texto com a política de confiança atualizada. É possível usar qualquer editor de texto para construir a política.

   Por exemplo, a seguinte política de confiança mostra como fazer referência a duas Contas da AWS no elemento `Principal`. Isso permite que os usuários de duas Contas da AWS separadas assumam esse perfil.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Se você especificar um principal em outra conta, adicionar uma conta à política de confiança de uma função é apenas metade da tarefa de estabelecer o relacionamento de confiança entre contas. Por padrão, nenhum usuário nas contas confiáveis pode assumir a função. O administrador da conta confiável recém-criada deve conceder aos usuários a permissão para assumir a função. Para fazer isso, o administrador deve criar ou editar uma política que está anexada ao usuário para permitir acesso ao usuário à ação `sts:AssumeRole`. Para obter mais informações, consulte o procedimento a seguir ou Conceder permissões a um usuário para alternar perfis.

4. Para usar o arquivo que você acabou de criar para atualizar a política de confiança, chame a seguinte operação:

   - UpdateAssumeRolePolicy

Para permitir que os usuários em uma conta externa confiável usem a função (API da AWS)

Para obter mais informações e detalhes sobre esse procedimento, consulte Conceder permissões a um usuário para alternar perfis.

1. Crie um arquivo JSON que contenha uma política de permissões que concede permissões para assumir a função. Por exemplo, a seguinte política contém as permissões necessárias mínimas:

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Action": "sts:AssumeRole",
           "Resource": "arn:aws:iam::111122223333:role/ROLE-NAME"
       }
   }
   ```

   Substitua o ARN na instrução pelo ARN da função que o usuário pode assumir.
2. Chame a seguinte operação para carregar o arquivo JSON que contém a política de confiança para o IAM:

   - CreatePolicy

   O resultado dessa operação inclui o ARN da política. Anote esse ARN, pois você precisará dele em uma etapa posterior.
3. Decida qual usuário ou grupo ao qual anexar a política. Se você não souber o nome do usuário ou do grupo pretendido, chame uma das seguintes operações para listar os usuários ou os grupos em sua conta:

   - ListUsers
   - ListGroups
4. Chame uma das seguintes operações para anexar a política criada na etapa anterior ao usuário ou ao grupo:

   - API: AttachUserPolicy

- AttachGroupPolicy

Atualizar permissões para um perfil

Use os procedimentos a seguir para atualizar as políticas de permissões e os limites de permissões de um perfil.

Pré-requisito: visualizar o acesso ao perfil

Antes de alterar as permissões de uma função, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Atualizar a política de permissões para um perfil

Para alterar as permissões permitidas pela função, modifique a política de permissões da função (ou políticas). Você não pode modificar a política de permissões para uma função vinculada ao serviço no IAM. Pode ser possível modificar a política de permissões no serviço que depende da função. Para verificar se um serviço oferece suporte a este recurso, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Funções vinculadas ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Atualizar a política de permissões de um perfil (console)

Para alterar as permissões permitidas por uma função (console)

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis.
3. Escolha o nome da função que você deseja modificar e, em seguida, escolha a guia Permissões.
4. Execute um destes procedimentos:

   - Para editar uma política gerenciada do cliente atual, escolha o nome da política e escolha Editar política.

> **Note**
>
> Você não pode editar uma política AWS gerenciada. As políticas AWS gerenciadas são exibidas com o ícone da AWS ( ). Para obter mais informações sobre a diferença entre políticas gerenciadas pela AWS e pelo cliente, consulte Políticas gerenciadas e em linha.

- Para anexar uma política gerenciada existente à função, escolha Add permissions (Adicionar permissões)e, depois, escolha Attach policies (Anexar políticas).
- Para editar uma política em linha existente, expanda a política e escolha Edit (Editar).
- Para incorporar uma nova política em linha, escolha Add permissions (Adicionar permissões) e, depois, escolha Create inline policy (Criar política em linha).
- Para remover uma política existente do perfil, marque a caixa de seleção ao lado do nome da política e escolha Remover.

Atualizar uma política de permissões de perfil (AWS CLI)

Para alterar as permissões permitidas pela função, modifique a política de permissões da função (ou políticas). Você não pode modificar a política de permissões para uma função vinculada ao serviço no IAM. Pode ser possível modificar a política de permissões no serviço que depende da função. Para verificar se um serviço oferece suporte a este recurso, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Funções vinculadas ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Para alterar as permissões permitidas por uma função (AWS CLI)

1. (Opcional) Para visualizar as permissões atuais associadas a uma função, execute um dos comandos a seguir:

   1. aws iam list-role-policies para listar as políticas em linha
   2. aws iam list-attached-role-policies para listar as políticas gerenciadas

2. O comando para atualizar as permissões para a função será diferente se você estiver atualizando uma política gerenciada ou uma política em linha.

Para atualizar uma política gerenciada, execute o seguinte comando para criar uma nova versão da política gerenciada:

- aws iam create-policy-version

Para atualizar uma política em linha, execute o seguinte comando:

- aws iam put-role-policy

Atualizar uma política de permissões de um perfil (API da AWS)

Para alterar as permissões permitidas pela função, modifique a política de permissões da função (ou políticas). Você não pode modificar a política de permissões para uma função vinculada ao serviço no IAM. Pode ser possível modificar a política de permissões no serviço que depende da função. Para verificar se um serviço oferece suporte a este recurso, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Funções vinculadas ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Para alterar as permissões permitidas por uma função (API da AWS)

1. (Opcional) Para visualizar as permissões atuais associadas a uma função, chame as seguintes operações:
   1. ListRolePolicies para listar as políticas em linha
   2. ListAttachedRolePolicies para listar as políticas gerenciadas
2. A operação para atualizar as permissões para a função será diferente se você estiver atualizando uma política gerenciada ou uma política em linha.

   Para atualizar uma política gerenciada, chame a seguinte operação para criar uma nova versão da política gerenciada:

   - CreatePolicyVersion

   Para atualizar uma política em linha, chame a seguinte operação:

   - PutRolePolicy

Atualizar o limite de permissões para um perfil

Para alterar o número máximo de permissões permitidas para uma função, modifique o limite de permissões da função.

Atualizar o limite de permissões de um perfil (console)

Para alterar a política usada para definir o limite de permissões para uma função

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Escolha o nome da função com o limite de permissões que você deseja alterar.
4. Escolha a aba Permissões. Se necessário, abra a seção Limite de permissões e, em seguida, escolha Alterar limite.
5. Selecione a política que você deseja usar para o limite de permissões.
6. Escolha Alterar limite.

   Suas alterações não terão efeito até que alguém assuma essa função.

Atualizar um limite de permissões de perfil (AWS CLI)

Para alterar a política gerenciada usada para definir o limite de permissões para uma função (AWS CLI)

1. (Opcional) Para visualizar a política de permissões atual de uma função, execute o seguinte comando:

   - aws iam get-role

2. Para usar uma política gerenciada diferente para atualizar o limite de permissões de uma função, execute o seguinte comando:

   - aws iam put-role-permissions-boundary

   Uma função pode ter apenas um conjunto de políticas gerenciadas como um limite de permissões. Se você alterar o limite de permissões, você altera o número máximo de permissões permitidas para uma função.

Atualizar um limite de permissões de perfil (API da AWS)

Para alterar a política gerenciada usada para definir o limite de permissões para uma função (API da AWS)

1. (Opcional) Para visualizar o limite de permissões atual de uma função, chame a seguinte operação:
   - GetRole
2. Para usar uma política gerenciada diferente para atualizar o limite de permissões de uma função, chame a seguinte operação:
   - PutRolePermissionsBoundary

   Uma função pode ter apenas um conjunto de políticas gerenciadas como um limite de permissões. Se você alterar o limite de permissões, você altera o número máximo de permissões permitidas para uma função.

Atualizar as configurações de um perfil

Use os procedimentos a seguir para atualizar a descrição de um perfil ou alterar a duração máxima da sessão de um perfil.

Atualizar uma descrição de perfil

Para alterar a descrição da função, modifique o texto da descrição.

Atualizar a descrição de um perfil (console)

Para alterar a descrição de uma função (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis.
3. Escolha o nome da função a ser modificada.
4. Na seção Summary (Resumo), escolha Edit (Editar).
5. Insira uma nova descrição na caixa e escolha Save changes (Salvar alterações).

Atualizar a descrição de um perfil (AWS CLI)

Para alterar a descrição de uma função (AWS CLI)

1. (Opcional) Para visualizar a descrição atual de uma função, execute o comando a seguir:
   - aws iam get-role
2. Para atualizar a descrição de uma função, execute o seguinte comando com o parâmetro de descrição:
   - aws iam update-role

Atualizar a descrição de um perfil (API da AWS)

Para alterar a descrição de uma função (API da AWS)

1. (Opcional) Para visualizar a descrição atual de uma função, chame a seguinte operação:
   - GetRole
2. Para atualizar a descrição de uma função, chame a seguinte operação com o parâmetro de descrição:
   - UpdateRole

Atualizar a duração máxima da sessão de um perfil

Para especificar a configuração de duração máxima da sessão para funções que são assumidas usando o console, a AWS CLI ou a API da AWS, modifique o valor da configuração da duração máxima da sessão. Essa configuração pode ter um valor de 1 hora a 12 horas. Se você não especificar um valor, o padrão máximo de 1 hora será aplicado. Essa configuração não limita sessões assumidas por serviços da AWS.

Atualizar a duração máxima da sessão de um perfil (console)

Como alterar a configuração de duração máxima da sessão para funções que são assumidas usando o console, a AWS CLI ou a API da AWS (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação do console do IAM, escolha Perfis.
3. Escolha o nome da função a ser modificada.
4. Na seção Summary (Resumo), escolha Edit (Editar).
5. Para Maximum session duration (Duração máxima da sessão), escolha um valor. Ou então, escolha Custom duration (Duração personalizada) e insira um valor (em segundos).
6. Escolha Salvar alterações.

   Suas alterações não terão efeito até que alguém assuma essa função. Para saber como revogar as sessões existentes para a função, consulte Revogar as credenciais de segurança temporárias do perfil do IAM.

No Console de gerenciamento da AWS, as sessões de usuário do IAM são de 12 horas por padrão. Os usuários do IAM que trocam de perfis no console recebem a duração máxima da sessão da perfil ou o tempo restante na sessão do usuário, o que for menor.

Qualquer pessoa que assuma a função da AWS CLI ou da API da AWS pode solicitar uma sessão mais longa, até esse máximo. A configuração `MaxSessionDuration` determina a duração máxima da sessão da função que pode ser solicitada.

- Para especificar a duração de uma sessão usando a AWS CLI, use o parâmetro `duration-seconds`. Para saber mais, consulte Alterar para uma perfil do IAM (AWS CLI).
- Para especificar a duração de uma sessão usando a API da AWS, use o parâmetro `DurationSeconds`. Para saber mais, consulte Alternar para um perfil do IAM (API da AWS).

Atualizar a duração máxima da sessão do perfil (AWS CLI)

> **Note**
>
> Qualquer pessoa que assuma uma função a partir da AWS CLI ou da API pode usar o parâmetro da CLI `duration-seconds` ou o parâmetro da API `DurationSeconds` para solicitar uma sessão mais longa. A configuração `MaxSessionDuration` determina a duração máxima da sessão da função que pode ser solicitada usando o parâmetro `DurationSeconds`. Se os usuários não especificarem um valor para o parâmetro `DurationSeconds`, suas credenciais de segurança serão válidas por uma hora.

Para alterar a configuração de duração máxima da sessão para funções que são assumidas usando a AWS CLI (AWS CLI)

1. (Opcional) Para visualizar a configuração de duração máxima da sessão atual para uma função, execute o seguinte comando:

   - aws iam get-role

2. Para atualizar uma configuração de duração máxima da sessão da função, execute o seguinte comando com o parâmetro da CLI `max-session-duration` ou o parâmetro da API `MaxSessionDuration`:

   - aws iam update-role

   Suas alterações não terão efeito até que alguém assuma essa função. Para saber como revogar as sessões existentes para a função, consulte Revogar as credenciais de segurança temporárias do perfil do IAM.

Atualizar a duração máxima da sessão do perfil (API da AWS)

> **Note**
>
> Qualquer pessoa que assuma uma função a partir da AWS CLI ou da API pode usar o parâmetro da CLI `duration-seconds` ou o parâmetro da API `DurationSeconds` para solicitar uma sessão mais longa. A configuração `MaxSessionDuration` determina a duração máxima da sessão da função que pode ser solicitada usando o parâmetro `DurationSeconds`. Se os usuários não especificarem um valor para o parâmetro `DurationSeconds`, suas credenciais de segurança serão válidas por uma hora.

Para alterar a configuração de duração máxima da sessão para funções que são assumidas usando a API (API da AWS)

1. (Opcional) Para visualizar a configuração de duração máxima da sessão atual para uma função, chame a seguinte operação:

   - GetRole

2. Para atualizar uma configuração de duração máxima da sessão da função, chame a seguinte operação com o parâmetro da CLI `max-session-duration` ou o parâmetro da API `MaxSessionDuration`:

   - UpdateRole

   Suas alterações não terão efeito até que alguém assuma essa função. Para saber como revogar as sessões existentes para a função, consulte Revogar as credenciais de segurança temporárias do perfil do IAM.

Excluir perfis ou perfis de instância

Se você não precisar mais de uma função, recomendamos excluir a função e suas permissões associadas. Dessa forma, você não terá uma entidade não utilizada que não seja ativamente monitorada ou mantida.

Se a função foi associada a uma instância do EC2, você também poderá remover a função do perfil de instância e excluir o perfil de instância.

> **Warning**
>
> Certifique-se de que você não tenha nenhuma instância do Amazon EC2 em execução com a função ou o perfil da instância que você está prestes a excluir. Excluir uma função ou perfil da instância associado a uma instância em execução interromperá todas as aplicações em execução na instância.

Se você preferir não excluir uma função permanentemente, poderá desabilitá-la. Para fazer isso, altere as políticas da função e revogue todas as sessões atuais. Por exemplo, você poderia adicionar uma política à função que negasse acesso a todas da AWS. Você também poderia editar a política de confiança para negar acesso a todos que tentassem assumir a função. Para obter mais informações sobre como revogar sessões, consulte Revogar as credenciais de segurança temporárias do perfil do IAM.

Tópicos

- Visualizar acesso a perfis
- Excluir um perfil vinculado ao serviço

- Exclusão de uma função do IAM (console)
- Exclusão de uma função do IAM (AWS CLI)
- Exclusão de uma função do IAM (API da AWS)
- Informações relacionadas

Visualizar acesso a perfis

Antes de excluir uma função, recomendamos que você revise quando a função foi usada pela última vez. Você pode fazer isso usando o Console de gerenciamento da AWS, a AWS CLI ou a API da AWS. Você deve visualizar essas informações porque não deseja remover o acesso de alguém que usa a função.

A data das atividades mais recentes do perfil pode não corresponder à última data informada na guia Acessados recentemente. A guia Acessados recentemente só informa as atividades em serviços permitidos pelas políticas de permissões do perfil. A data da última atividade da função inclui a última tentativa de acessar qualquer produto na AWS.

> **Note**
>
> O período de rastreamento das atividades mais recentes de um perfil e os dados de Acessados recentemente são para os últimos 400 dias. Esse período pode ser mais curto se a sua Região começou a oferecer suporte a esses recursos no último ano. A função pode ter sido usada há mais de 400 dias. Para obter mais informações sobre o período de rastreamento, consulte Onde a AWS rastreia informações acessadas por último.

Como visualizar quando uma função foi usada pela última vez (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Localize a linha da função com a atividade que você deseja visualizar. É possível usar o campo de pesquisa para restringir os resultados. Exiba a coluna Last activity (Última atividade) para visualizar o número de dias desde que a função foi usada pela última vez. Se a função não tiver sido usada dentro do período de rastreamento, a tabela exibirá None (Nenhum).
4. Escolha o nome da função para exibir mais informações. A página Summary (Resumo) da função também inclui a Last activity (Última atividade), que exibe a data em que a função foi

usada pela última vez. Se a função não tiver sido usada nos últimos 400 dias, a Last activity (Última atividade) exibirá Não acessada no período de rastreamento.

Como visualizar quando uma função foi usada pela última vez (AWS CLI)

`aws iam get-role` – execute este comando para receber informações sobre uma função, incluindo o objeto `RoleLastUsed`. Este objeto contém a `LastUsedDate` e a `Region` em que a função foi usada pela última vez. Se `RoleLastUsed` estiver presente, mas não contiver um valor, a função não foi usada dentro do período de rastreamento.

Como visualizar quando uma função foi usada pela última vez (API da AWS)

`GetRole` – chame esta operação para retornar informações sobre uma função, incluindo o objeto `RoleLastUsed`. Este objeto contém a `LastUsedDate` e a `Region` em que a função foi usada pela última vez. Se `RoleLastUsed` estiver presente, mas não contiver um valor, a função não foi usada dentro do período de rastreamento.

Excluir um perfil vinculado ao serviço

O método que você usa para excluir um perfil vinculado ao serviço depende do serviço. Em alguns casos, você não precisa excluir manualmente uma função vinculada ao serviço. Por exemplo, quando você concluir uma ação específica (como a remoção de um recurso) no serviço, o serviço pode excluir a função vinculada ao serviço para você. Em outros casos, o serviço pode oferecer suporte a exclusão de uma função vinculada ao serviço manualmente usando o console, API ou a do serviço AWS CLI.

Revise a documentação para o perfil vinculado ao serviço no serviço vinculado para saber como excluir o perfil. Você pode visualizar as funções vinculadas ao serviço na sua conta acessando a página Roles (Funções) do IAM no console. As funções vinculadas ao serviço aparecem com (Função vinculada ao serviço) na coluna Entidades confiáveis da tabela. Um banner na página Summary (Resumo) da função também indica que a função é uma função vinculada ao serviço.

Se o serviço não incluir documentação para excluir a função vinculada ao serviço, você poderá usar o console do IAM, a AWS CLI ou a API para excluir a função.

Exclusão de uma função do IAM (console)

Ao usar o Console de gerenciamento da AWS para excluir um perfil, o IAM desvincula automaticamente as políticas associadas ao perfil. Também exclui automaticamente as políticas em linha associadas ao perfil e todos os perfis de instâncias do Amazon EC2 que contêm o perfil.

> **Important**
>
> Em alguns casos, uma função pode ser associada a um perfil de instância do Amazon EC2, e a função e o perfil de instância podem ter o mesmo nome. Neste caso, você pode usar o Console de gerenciamento da AWS para excluir a função e o perfil da instância. Essa ligação ocorre automaticamente para funções e perfis de instância criados no console. Se você criou a função da AWS CLI, do Tools for Windows PowerShell ou da API da AWS, a função e o perfil da instância poderão ter nomes diferentes. Nesse caso, você não pode usar o console para excluí-los. Em vez disso, você deve usar a AWS CLI, o Tools for Windows PowerShell ou a API da AWS para remover primeiro a função do perfil da instância. Você deve executar uma etapa separada para excluir a função.

Para excluir uma função (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Roles (Funções) e marque a caixa de seleção ao lado do nome da função que você deseja excluir.
3. Na parte superior da página, escolha Delete (Excluir).
4. Na caixa de diálogo de confirmação, revise as informações acessadas por último, que mostram quando cada uma das funções selecionadas acessou pela última vez um serviço da AWS. Isso ajuda você a confirmar se a função está ativa no momento. Se você quiser continuar, insira o nome da função no campo de entrada de texto e escolha Delete (Excluir). Se você tiver certeza, prossiga com a exclusão, mesmo se as informações acessadas por último ainda estiverem sendo carregadas.

> **Note**
>
> Você não pode usar o console para excluir um perfil de instância, a menos que ele tenha o mesmo nome da função. O perfil da instância é excluído como parte do processo de exclusão de uma função, conforme descrito no procedimento anterior. Para excluir um perfil de instância sem também excluir a função, você deve usar a AWS CLI, ou a API da AWS. Para obter mais informações, consulte as seções a seguir.

Exclusão de uma função do IAM (AWS CLI)

Ao usar a AWS CLI para excluir uma função, primeiro é necessário excluir as políticas em linha associadas à função. Também é necessário separar as políticas gerenciadas associadas ao perfil. Se quiser excluir o perfil de instância associado que contém o perfil, exclua-o separadamente.

Para excluir uma função (AWS CLI)

1. Se você não souber o nome da função que deseja excluir, digite o seguinte comando para listar as funções em sua conta:

   ```
   aws iam list-roles
   ```

   A lista inclui o nome de recurso da Amazon (ARN) de cada função. Use o nome da função, não o nome de recurso da Amazon (ARN), para fazer referência às funções com os comandos da CLI. Por exemplo, se uma função tiver o seguinte nome de recurso da Amazon (ARN): `arn:aws:iam::123456789012:role/myrole`, você fará referência à função como **`myrole`**.
2. Remova o perfil de todos os perfis de instância aos quais ele está associado.

   a. Para relacionar todos os perfis de instância aos quais a função está associada, digite o seguinte comando:

      ```
      aws iam list-instance-profiles-for-role --role-name role-name
      ```

   b. Para remover a função de um perfil de instância, digite o seguinte comando para cada perfil de instância:

      ```
      aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
      ```
3. Exclua todas as políticas associadas à função.

   a. Para listar todas as políticas em linha no perfil, digite o seguinte comando:

      ```
      aws iam list-role-policies --role-name role-name
      ```

   b. Para excluir cada política em linha do perfil, digite o comando a seguir para cada política:

      ```
      aws iam delete-role-policy --role-name role-name --policy-name policy-name
      ```

c. Para listar todas as políticas gerenciadas que estão associadas ao perfil, digite o seguinte comando:

```
aws iam list-attached-role-policies --role-name role-name
```

d. Para desassociar cada política gerenciada do perfil, digite o comando a seguir para cada política:

```
aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
```

4. Digite o comando a seguir para excluir a função:

```
aws iam delete-role --role-name role-name
```

5. Se você não planeja reutilizar o perfis de instância que foram associados à função, digite o seguinte comando para excluí-los:

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Exclusão de uma função do IAM (API da AWS)

Ao usar a API do IAM para excluir uma função, primeiro é necessário excluir as políticas associadas à função. Também é necessário separar as políticas gerenciadas associadas ao perfil. Se quiser excluir o perfil de instância associado que contém o perfil, exclua-o separadamente.

Para excluir uma função (API da AWS)

1. Para listar todos os perfis de instância aos quais um perfil está associado, chame ListInstanceProfilesForRole.

   Para remover o perfil de um perfil de instância, chame RemoveRoleFromInstanceProfile. É necessário transmitir o nome do perfil e nome do perfil de instância.

   Se você não for reutilizar um perfil de instância associado à função, chame DeleteInstanceProfile para excluí-lo.

2. Para listar todas as políticas em linha para um perfil, chame ListRolePolicies.

   Para excluir políticas em linha associadas ao perfil, chame DeleteRolePolicy. É necessário passar o nome do perfil e o nome da política em linha.

3. Para listar todas as políticas gerenciadas associadas a um perfil, chame ListAttachedRolePolicies.

   Para desassociar políticas gerenciadas associadas ao perfil, chame DetachRolePolicy. É necessário passar o nome do perfil e o ARN da política gerenciada.

4. Chame DeleteRole para excluir a função.

Informações relacionadas

Para obter informações gerais sobre os perfis de instância, consulte Usar perfis de instância.

Para obter informações gerais sobre funções vinculadas ao serviço, consulte Criar um perfil vinculado ao serviço.

Métodos para assumir um perfil

Para que um usuário, uma aplicação ou um serviço possa usar um perfil que você criou, você deverá conceder permissões para alternar para esse perfil. É possível usar qualquer política anexada a grupos ou usuários para conceder as permissões necessárias. Depois que as permissões forem concedidas, o usuário poderá assumir um perfil do Console de gerenciamento da AWS, do Tools for Windows PowerShell, da AWS Command Line Interface (AWS CLI) e da API `AssumeRole`.

> **Important**
>
> Ao criar uma função de forma programática, em vez de no console do IAM, você tem a opção de adicionar um `Path` de até 512 caracteres além do `RoleName`, que pode ter até 64 caracteres. No entanto, se você pretende usar uma função com o recurso Switch Role (Alternar função) no Console de gerenciamento da AWS, o `Path` e o `RoleName` combinados não podem exceder 64 caracteres.

O método usado para assumir o perfil determina quem pode assumir a função e por quanto tempo a sessão da função pode durar. Ao usar o código `AssumeRole*` das operações da API, o perfil do IAM que você assume é o de recursos. O usuário ou perfil que chama operações da API `AssumeRole*` é a entidade principal.

A tabela a seguir compara os métodos usados para assumir perfis.

Método de assumir a função	Quem pode assumir a função	Método para especificar a vida útil da credencial	Vida útil da credencial (mín.|máx.|padrão)
Console de gerenciamento da AWS	Usuário ou perfis[1] (via alternância de perfis)	Maximum session duration (Duração máxima da sessão) na página Resumo Role (Função)	15 min|Configuração de duração máxima da sessão[2]|1 h
Operação da CLI `assume-role` ou da API `AssumeRole`	Usuário ou perfil[1]	Parâmetro da CLI `duration-seconds` ou da API `DurationSeconds`	15 min|Configuração de duração máxima da sessão[2]|1 h
Operação da CLI `assume-role-with-saml` ou da API `AssumeRoleWithSAML`	Qualquer usuário autenticado usando SAML	Parâmetro da CLI `duration-seconds` ou da API `DurationSeconds`	15 min|Configuração de duração máxima da sessão[2]|1 h
Operação da CLI `assume-role-with-web-identity` ou da API `AssumeRoleWithWebIdentity`	Qualquer usuário autenticado usando um provedor OIDC	Parâmetro da CLI `duration-seconds` ou da API `DurationSeconds`	15 min|Configuração de duração máxima da sessão[2]|1 h

Método de assumir a função	Quem pode assumir a função	Método para especificar a vida útil da credencial	Vida útil da credencial (mín.|máx.|padrão)
URL do console construído com `AssumeRole`	Usuário ou perfil	Parâmetro HTML `SessionDuration` no URL	15 min|12 h|1 h
URL do console construído com `AssumeRoleWithSAML`	Qualquer usuário autenticado usando SAML	Parâmetro HTML `SessionDuration` no URL	15 min|12 h|1 h
URL do console construído com `AssumeRoleWithWebIdentity`	Qualquer usuário autenticado usando um provedor OIDC	Parâmetro HTML `SessionDuration` no URL	15 min|12 h|1 h

[1] O uso de credenciais de um perfil para assumir um perfil diferente é chamado de encadeamento de perfis. Quando você usa o encadeamento de perfis, a duração da sessão do perfil é limitada a uma hora. Isso se aplica à alternância de perfis do Console de gerenciamento da AWS, à AWS CLI e às operações de API. Essa limitação não se aplica à situação em que um perfil é assumido pela primeira vez por credenciais do usuário nem a aplicações executadas em instâncias do Amazon EC2 usando perfis de instância.

[2] Essa configuração pode ter um valor de 1 hora a 12 horas. Para obter detalhes sobre como modificar a configuração de duração máxima da sessão, consulte Gerenciamento de perfis do IAM. Essa configuração determina a duração máxima da sessão que você pode solicitar ao obter as credenciais da função. Por exemplo, quando você usa as operações da API AssumeRole* para assumir uma função, você pode especificar um tamanho de sessão usando o parâmetro `DurationSeconds`. Use este parâmetro para especificar o tamanho da sessão da função de 900 segundos (15 minutos) até o valor configurado da duração máxima da sessão para a função. Os usuários do IAM que trocam de perfis no console recebem a duração máxima da sessão ou o tempo restante na sessão de usuário, o que for menor. Suponha que você defina uma duração máxima de 5

horas em uma função. Um usuário do IAM conectado ao console por 10 horas (do máximo padrão de 12) alterna para a função. A duração da sessão de função disponível é de 2 horas. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil mais adiante nesta página.

Observações

- A configuração da duração máxima da sessão não limita as sessões assumidas por produtos da AWS.
- As credenciais do perfil do IAM do Amazon EC2 não estão sujeitas à duração máxima da sessão configurada no perfil.
- Para permitir que os usuários assumam novamente o perfil atual em uma sessão de perfil, especifique o ARN do perfil ou o ARN da Conta da AWS como entidade principal na política de confiança do perfil. Os Serviços da AWS que fornecem recursos computacionais, como o Amazon EC2, Amazon ECS, Amazon EKS e Lambda, fornecem credenciais temporárias e atualizam automaticamente essas credenciais. Isso garante que você tenha sempre um conjunto de credenciais válido. Nesses serviços, não é necessário assumir novamente a função atual para obter credenciais temporárias. Porém, se pretender passar tags de sessão ou uma política de sessão, você precisará assumir novamente a função atual. Para saber como modificar uma política de confiança de função para adicionar o ARN da função de entidade principal ou o ARN da Conta da AWS, consulte Atualizar a política de confiança de um perfil.

Tópicos

- Mudar de um usuário para um perfil do IAM (console)
- Alterar para uma perfil do IAM (AWS CLI)
- Alternar para um perfil do IAM (Ferramentas para Windows PowerShell)
- Alternar para um perfil do IAM (API da AWS)
- Use um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2
- Usar perfis de instância

Mudar de um usuário para um perfil do IAM (console)

Você pode alternar perfis quando fizer login como usuário do IAM, como usuário no Centro de Identidade do IAM, como perfil federado SAML ou como perfil federado de identidade da Web. Uma função especifica um conjunto de permissões que você pode usar para acessar os recursos da AWS de que você precisa. No entanto, você não inicia sessão em um perfil, mas, depois de entrar como usuário do IAM, poderá mudar para um perfil do IAM. Isso separa, temporariamente, as permissões originais de usuário e, em vez disso, oferece a você as permissões atribuídas à função. O perfil pode estar em sua própria conta ou em qualquer outra Conta da AWS. Para obter mais informações sobre funções, seus benefícios e como criá-los, consulte Perfis do IAMe Criar um perfil do IAM.

As permissões do seu usuário e quaisquer perfis para os quais você mudar não são cumulativos. Apenas um conjunto de permissões é ativo por vez. Quando você muda para uma função, perde, temporariamente, as permissões de usuário e trabalha com permissões atribuídas à função. Ao sair da função, suas permissões de usuário são, automaticamente, restauradas.

Quando você muda de funções no Console de gerenciamento da AWS, o console sempre usa suas credenciais originais para autorizar a mudança. Por exemplo, se você alternar para RoleA, o IAM usará suas credenciais originais para determinar se você tem permissão para assumir RoleA. Se você tentar mudar para RoleB enquanto usa RoleA, o AWS ainda usará suas credenciais originais para autorizar a mudança, e não as credenciais de RoleA.

> **Note**
>
> Ao iniciar sessão como um usuário no Centro de Identidade do IAM, seja como um perfil federado SAML ou perfil federado de identidade da Web, você assumirá um perfil do IAM ao iniciar sua sessão. Por exemplo, quando um usuário no IAM Identity Center faz login no portal de acesso da AWS, pode escolher um conjunto de permissões ou acessar um perfil atribuído por meio do gerenciador de acesso à conta antes de poder acessar os recursos da AWS.

Sessões de função

Ao mudar de perfis, sua sessão do Console de gerenciamento da AWS dura 1 hora por padrão. As sessões de usuário do IAM são de 12 horas por padrão. Outros usuários podem ter durações de sessão diferentes definidas. Quando você muda de perfil no console, recebe a duração máxima da sessão ou o tempo restante na sessão de usuário, o que for menor. Ao assumir um perfil, não é possível pode estender a duração da sessão.

Por exemplo, suponha que uma duração máxima de sessão de dez horas seja definida para uma função. Você estava conectado ao console por 8 horas quando decide mudar para o perfil. Há 4 horas restantes em sua sessão de usuário. Portanto, a duração permitida da sessão de perfil é 4 horas, e não a duração máxima de 10 horas da sessão. A tabela a seguir mostra como determinar a duração da sessão de um usuário do IAM ao alternar funções no console.

O tempo restante da sessão do usuário do IAM é de…	A duração da sessão da função é de…
Menor que a duração máxima da sessão da função	O tempo restante na sessão do usuário
Maior que a duração máxima da sessão da função	Valor de duração máxima da sessão
Igual à duração máxima da sessão da função	Valor máximo da duração da sessão (aproximado)

O uso de credenciais de um perfil para assumir um perfil diferente é chamado de encadeamento de perfis. Quando você usa o encadeamento de perfis, a duração da sessão é limitada a uma hora, independentemente da configuração de duração máxima da sessão definida para perfis individuais. Isso se aplica à alternância de perfis do Console de gerenciamento da AWS, à AWS CLI e às operações de API.

> **Note**
>
> Alguns consoles de produtos da AWS podem renovar automaticamente sua sessão de função quando ela expira sem que você execute nenhuma ação. Alguns podem solicitar que você recarregue a página do navegador para autenticar novamente sua sessão.

Considerações

- Você não poderá mudar de perfis caso se conecte como Usuário raiz da conta da AWS.
- Os usuários devem receber permissão para mudar de perfil por meio de políticas. Para instruções, consulte Conceder permissões a um usuário para alternar perfis.
- Não é possível mudar de perfil no Console de gerenciamento da AWS para um perfil que exija um valor ExternalId. Você só pode alternar para tal perfil chamando a API `AssumeRole`, que é compatível com o parâmetro `ExternalId`.

Para mudar para uma função

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Sign in to the Console de gerenciamento da AWS no Início de Sessão da AWS User Guide.
2. No Console de gerenciamento da AWS, escolha seu nome de usuário na barra de navegação no canto superior direito. Geralmente, ele é assim: ***username***@***account_ID_number_or_alias***.
3. Selecione um dos seguintes métodos para alternar perfis:
   - Selecione Alternar perfil.
   - Se você optou pela compatibilidade com várias sessões, escolha Adicionar sessão e selecione Alternar perfil.

   > **Note**
   >
   > Você pode fazer login com até cinco identidades diferentes simultaneamente em um único navegador da Web no Console de gerenciamento da AWS. Para obter detalhes, consulte Signing in to multiple accounts no Guia de conceitos básicos do Console de gerenciamento da AWS.

4. Na página Mudar de função, digite o número do ID da conta ou o alias da conta e o nome da função fornecida pelo administrador.

   > **Note**
   >
   > Se o administrador criou a função com um caminho, como `division_abc/subdivision_efg/roleToDoX`, será necessário digitar este caminho completo e o nome na caixa Função. Se você digitar apenas o nome da função, ou se o `Path` e o `RoleName` combinados excederem 64 caracteres, a mudança de função falhará. Este é um limite dos cookies do navegador que armazena o nome da função. Se isso acontecer, entre em contato com o administrador e peça para reduzir o tamanho do nome do caminho e da função.

5. (Opcional) É possível inserir um nome de exibição e selecionar uma cor de exibição que destacará a função na barra de navegação do console.

   - Em Display name, digite o texto que deseja que seja exibido na barra de navegação no lugar de seu nome de usuário quando este perfil estiver ativo. Um nome é sugerido, baseado nas informações da conta e da função, mas você poderá alterá-lo para outro com significado para você.
   - Em Display color, selecione uma cor para destacar o nome de exibição.

   O nome e a cor podem ajudar a lembrá-lo quando esta função está ativa, o que muda suas permissões. Por exemplo, para uma função que oferece acesso ao ambiente de teste, você pode especificar um Display Name (Nome de exibição) de **`Test`** e selecionar verde em Color (Cor). Para a função que oferece acesso à produção, você pode especificar um Display Name (Nome de exibição) de **`Production`** e selecionar vermelho em Color (Cor).

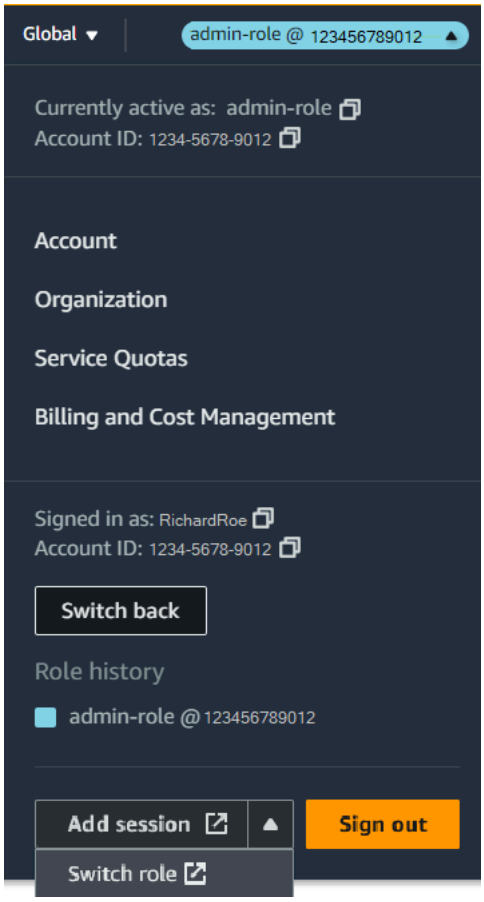
6. Selecione Mudar de perfil. O nome de exibição e a cor substituem seu nome de usuário na barra de navegação e você pode começar a usar as permissões concedidas pela função.

7. Depois de concluir as tarefas que exigem o perfil do IAM, você poderá voltar para a sessão original. Isso removerá as permissões adicionais fornecidas pelo perfil e restaurará as suas permissões padrão.

   a. No console do IAM, escolha o Display Name (Nome de exibição) da sua função na barra de navegação no canto superior direito.

   b. Selecione Mudar de volta.

Por exemplo, suponha que você tenha feito login à conta número `123456789012` usando o nome de usuário `Richard`. Depois de usar a função `admin-role`, você deseja interromper o uso da função e retornar para suas permissões originais. Para parar de usar o perfil, escolha admin-role @ 123456789012 e, em seguida, escolha Mudar de volta.



> **Tip**
>
> As últimas funções que você usou aparecem no menu. Na próxima vez que você precisar mudar para um desses perfis, bastará escolher o perfil desejado. Você só precisará digitar as informações da conta e do perfil manualmente se o perfil não for exibido no menu.

Recursos adicionais

- Conceder permissões a um usuário para alternar perfis
- Conceda permissões a um usuário para passar um perfil para um serviço da AWS

- Criar um perfil para conceder permissões a um usuário do IAM
- Criar um perfil para delegar permissões a um serviço da AWS
- Solucionar problemas de perfis do IAM

Alterar para uma perfil do IAM (AWS CLI)

Uma função especifica um conjunto de permissões que você pode usar para acessar os recursos da AWS de que você precisa. Nesse sentido, ela é semelhante a um usuário do AWS Identity and Access Management (IAM). Ao fazer login como usuário, você obtém um conjunto específico de permissões. No entanto, você não faz login em uma função mas, depois de fazer login como usuário, poderá mudar para uma função. Isso separa, temporariamente, as permissões originais de usuário e, em vez disso, oferece a você as permissões atribuídas à função. O perfil pode estar em sua própria conta ou em qualquer outra Conta da AWS. Para obter mais informações sobre funções, seus benefícios e como criar e configurá-las, consulte Perfis do IAM e Criar um perfil do IAM. Para saber mais sobre os diferentes métodos que você pode usar para assumir uma função, consulte Métodos para assumir um perfil.

> **Important**
>
> As permissões do usuário do IAM e quaisquer funções você venha a assumir não são cumulativas. Apenas um conjunto de permissões é ativo por vez. Quando você assume um perfil, perde temporariamente as permissões de usuário ou perfil anteriores e trabalha com as permissões atribuídas ao perfil. Ao sair da função, suas permissões de usuário são, automaticamente, restauradas.

Você poderá usar uma função para executar um comando da AWS CLI quando estiver conectado como um usuário do IAM. Você também pode usar um perfil para executar um comando da AWS CLI quando tiver se conectado como usuário autenticado externamente (SAML ou OIDC) que já esteja usando um perfil. Além disso, você pode usar uma função para executar um comando da AWS CLI de uma instância do Amazon EC2 anexada a uma função por meio do perfil da instância. Você não pode assumir um perfil quando está conectado como o Usuário raiz da conta da AWS.

Encadeamento de funções: você também pode usar o encadeamento de funções, que usa permissões de uma função para acessar uma segunda função.

Por padrão, a sessão da função dura uma hora. Quando assume esta função usando as operações da CLI `assume-role*`, você pode especificar um valor para o parâmetro `duration-seconds`.

Esse valor pode variar de 900 segundos (15 minutos) até o valor configurado de duração máxima da sessão para o perfil. Se você alternar perfis no console, a duração da sessão será limitada a, no máximo, uma hora. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil.

Se você usar o encadeamento de funções, a duração da sessão será limitada a um máximo de uma hora. Se você usar o parâmetro `duration-seconds` para fornecer um valor maior do que uma hora, a operação falhará.

Cenário de exemplo: alternar para uma função de produção

Imagine que você seja um usuário do IAM para trabalhar no ambiente de desenvolvimento. Nesse cenário, ocasionalmente, é necessário trabalhar com o ambiente de produção na linha de comando com a AWS CLI. Você já tem uma credencial de chave de acesso disponível para você. Esse pode ser o par de chaves de acesso atribuído ao usuário do IAM padrão. Ou, se estiver conectado como uma entidade principal federada SAML ou OIDC, ele poderá ser o par de chaves de acesso para o perfil inicialmente atribuído a você. Se suas permissões atuais concederem a capacidade de assumir uma função específica do IAM, você poderá identificar essa função em um "perfil" nos arquivos de configuração da AWS CLI. Esse comando é executado com as permissões da função do IAM especificada, não a identidade original. Quando especifica esse perfil em um comando da AWS CLI, você está usando a nova função. Nesta situação, você não pode usar as permissões originais na conta de desenvolvimento ao mesmo tempo. Isso porque apenas um conjunto de permissões pode ser ativado por vez.

> **Note**
>
> Por motivos de segurança, os administradores podem revisar logs do AWS CloudTrail para saber quem executou uma ação na AWS. Seu administrador pode exigir que você especifique uma identidade-fonte ou um nome de sessão de função ao assumir a função. Para obter mais informações, consulte `sts:SourceIdentity` e `sts:RoleSessionName`.

Para alternar para uma função de produção (AWS CLI)

1. Se você nunca usou a AWS CLI, é necessário, primeiro, configurar seu perfil CLI padrão. Abra um prompt de comando e configure sua instalação da AWS CLI para usar a chave de acesso de seu usuário do IAM ou de sua função federada. Para obter mais informações, consulte Configuração da AWS Command Line Interface no Guia do usuário da AWS Command Line Interface.

Execute o comando aws configure da seguinte forma:

```
aws configure
```

Quando solicitado, forneça as seguintes informações:

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-east-2
Default output format [None]: json
```

2. Crie um novo perfil para a função no arquivo `.aws/config` no Unix ou Linux ou o arquivo `C:\Users\USERNAME\.aws\config` no Windows. O exemplo a seguir cria um perfil chamado `prodaccess` que muda para a função *`ProductionAccessRole`* na conta `123456789012`. Você obtém o ARN da função do administrador da conta que criou a função. Quando este perfil é invocado, a AWS CLI usa as credenciais do `source_profile` para solicitar credenciais para a função. Por isso, a identidade mencionada como `source_profile` deve ter `sts:AssumeRole` permissões para a função especificada no `role_arn`.

```
[profile prodaccess]
    role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
    source_profile = default
```

3. Depois de criar o novo perfil, qualquer comando da AWS CLI que especifique o parâmetro `--profile prodaccess` será executado sob as permissões anexadas à função `ProductionAccessRole` do IAM, em vez do usuário padrão.

```
aws iam list-users --profile prodaccess
```

Este comando funciona se as permissões atribuídas ao `ProductionAccessRole` permitem a listagem dos usuários na conta da AWS atual.

4. Para retornar para as permissões concedidas por suas credenciais originais, execute comandos sem o parâmetro `--profile`. A AWS CLI reverte para usar as credenciais em seu perfil padrão, que você configurou em Step 1.

Para obter mais informações, consulte Assumir uma função no Guia do usuário do AWS Command Line Interface.

Cenário de exemplo: permitir que uma função de perfil da instância alterne para uma função em outra conta

Imagine que você esteja usando duas Contas da AWS e queira permitir que uma aplicação em execução em uma instância do Amazon EC2 execute comandos da AWS CLI nas duas contas. Vamos supor que a instância do EC2 exista na conta `111111111111`. Essa instância inclui a função de perfil da instância `abcd` que permite que a aplicação execute tarefas somente leitura do Amazon S3 no bucket `amzn-s3-demo-bucket1` dentro da mesma conta `111111111111`. No entanto, o aplicativo também deve ter permissão para assumir a função entre contas `efgh` para executar tarefas na conta `222222222222`. Para fazer isso, a função de perfil de instância do EC2 `abcd` deve ter a seguinte política de permissões:

Política de permissões da função ***abcd*** da conta 111111111111

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
```

```
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Vamos supor que a função entre contas `efgh` permita tarefas somente leitura do Amazon S3 no bucket `amzn-s3-demo-bucket2` na mesma conta `222222222222`. Para fazer isso, a função entre contas `efgh` deve ter a seguinte política de permissões:

Política de permissões da função ***efgh*** da conta 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
```

```
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

A função `efgh` deve permitir que a função do perfil de instância `abcd` a assuma. Para fazer isso, a função `efgh` deve ter a seguinte política de confiança:

Política de confiança da função ***efgh*** da conta 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Para executar comandos da AWS CLI na conta `222222222222`, você deve atualizar o arquivo de configuração da CLI. Identifique a função `efgh` como "perfil" e a função do perfil de instância do EC2 `abcd` como a “fonte de credencial” no arquivo de configuração da AWS CLI. Os comandos da CLI são executados com as permissões da função `efgh`, e não a função `abcd` original.

> **Note**
>
> Para fins de segurança, você pode usar o AWS CloudTrail para auditar o uso de funções na conta. Para diferenciar sessões de função quando uma função é usada por diferentes entidades de segurança nos logs do CloudTrail, use o nome da sessão da função. Quando a AWS CLI assume uma função em nome de um usuário, como descrito neste tópico, um nome de sessão da função é automaticamente criado como `AWS-CLI-session-`*nnnnnnnn*. O valor *nnnnnnnn* é um número inteiro que representa o tempo no formato de horário epoch

Unix (o número de segundos desde a meia-noite de 1º de janeiro de 1970, em UTC). Para obter mais informações, consulte Referência de eventos do CloudTrail no Guia do usuário do AWS CloudTrail.

Para permitir que uma função de perfil de instância do EC2 alterne para uma função entre contas (AWS CLI)

1. Você não precisa configurar um perfil de CLI padrão. Em vez disso, você pode carregar credenciais dos metadados de perfil da instância do EC2. Crie um novo perfil para a função no arquivo `.aws/config`. O exemplo a seguir cria um perfil `instancecrossaccount` que muda para a função *`efgh`* na conta `222222222222`. Quando esse perfil é invocado, a AWS CLI usa as credenciais dos metadados de perfil da instância do EC2 para solicitar credenciais para a função. Por isso, a função de perfil de instância do EC2 deve ter permissões `sts:AssumeRole` para a função especificada no `role_arn`.

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. Depois de criar o novo perfil, qualquer comando da AWS CLI que especifique o parâmetro `--profile instancecrossaccount` é executado sob as permissões anexadas à função `efgh` na conta `222222222222`.

   ```
   aws s3 ls amzn-s3-demo-bucket2 --profile instancecrossaccount
   ```

   Esse comando funciona se as permissões atribuídas ao perfil `efgh` permitem a listagem dos usuários na Conta da AWS atual.

3. Para retornar para as permissões originais do perfil de instância do EC2 na conta `111111111111`, execute os comandos da CLI sem o parâmetro `--profile`.

Para obter mais informações, consulte Assumir uma função no Guia do usuário do AWS Command Line Interface.

Alternar para um perfil do IAM (Ferramentas para Windows PowerShell)

Uma função especifica um conjunto de permissões que você pode usar para acessar os recursos da AWS de que você precisa. Nesse sentido, ela é semelhante a um usuário do AWS Identity and

Access Management (IAM). Ao fazer login como usuário, você obtém um conjunto específico de permissões. No entanto, você não faz login em uma função, mas uma vez que fez login, pode mudar para uma função. Isso separa, temporariamente, as permissões originais de usuário e, em vez disso, oferece a você as permissões atribuídas à função. O perfil pode estar em sua própria conta ou em qualquer outra Conta da AWS. Para obter mais informações sobre funções, seus benefícios e como criar e configurá-las, consulte Perfis do IAM e Criar um perfil do IAM.

> **Important**
>
> As permissões do seu usuário do IAM e quaisquer funções para as quais você mude não são cumulativas. Apenas um conjunto de permissões é ativo por vez. Quando você muda para uma função, perde, temporariamente, as permissões de usuário e trabalha com permissões atribuídas à função. Ao sair da função, suas permissões de usuário são, automaticamente, restauradas.

Esta seção descreve como alternar entre funções ao trabalhar na linha de comando com o AWS Tools for Windows PowerShell.

Imagine que você tenha uma conta no ambiente de desenvolvimento e, ocasionalmente, precise trabalhar com o ambiente de produção na linha de comando usando o Tools for Windows PowerShell. Você já tem um conjunto de credenciais de chave de acesso disponível para você. Elas podem ser um par de chaves de acesso atribuído ao seu usuário padrão do IAM. Ou, se você tiver feito login como uma entidade principal federada SAML ou OIDC, elas poderão ser o par de chaves de acesso para o perfil inicialmente atribuído a você. Você pode usar essas credenciais para executar o cmdlet `Use-STSRole` que transmite o ARN de uma nova função como um parâmetro. O comando retorna credenciais de segurança temporárias para a função solicitada. Em seguida, você pode usar essas credenciais em comandos subsequentes do PowerShell com as permissões da função para acessar recursos na produção. Enquanto você usa a função, não pode usar suas permissões de usuário na conta de desenvolvimento, pois apenas um conjunto de permissões pode estar ativado por vez.

> **Note**
>
> Por motivos de segurança, os administradores podem revisar logs do AWS CloudTrail para saber quem executou uma ação na AWS. Seu administrador pode exigir que você

especifique uma identidade-fonte ou um nome de sessão de função ao assumir a função. Para obter mais informações, consulte `sts:SourceIdentity` e `sts:RoleSessionName`.

Todas as chaves de acesso e tokens são apenas exemplos e não podem ser usados da forma que são mostrados. Substitua pelos valores apropriados do seu ambiente real.

Para alternar para uma função (Tools for Windows PowerShell)

1. Abra um prompt de comando do PowerShell e configure o perfil padrão para usar a chave de acesso do usuário do IAM ou de sua função federada. Se você já tive usado o Tools for Windows PowerShell, isso provavelmente já foi feito. Será possível alternar entre perfis somente se você estiver conectado como um usuário do IAM, e não como Usuário raiz da conta da AWS.

   ```
   PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
   SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
   PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
   ```

   Para obter mais informações, consulte Uso de credenciais da AWS no Guia do usuário do Ferramentas da AWS para PowerShell.

2. Para recuperar credenciais para a nova função, execute o comando a seguir para alternar para a função *RoleName* na conta 123456789012. Você obtém o ARN da função do administrador da conta que criou a função. O comando exige que você também forneça um nome de sessão. Você pode escolher qualquer texto para isso. O comando a seguir solicita as credenciais e, em seguida, captura o objeto de propriedade `Credentials` do objeto de resultados retornados e o armazena na variável `$Creds`.

   ```
   PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
   RoleSessionName "MyRoleSessionName").Credentials
   ```

   `$Creds` é um objeto que agora contém os elementos `AccessKeyId`, `SecretAccessKey` e `SessionToken` de que você precisa nas etapas a seguir. Os seguintes comandos de exemplo ilustram valores comuns:

   ```
   PS C:\> $Creds.AccessKeyId
   AKIAIOSFODNN7EXAMPLE

   PS C:\> $Creds.SecretAccessKey
   ```

```
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

PS C:\> $Creds.SessionToken
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Para usar essas credenciais para qualquer comando subsequente, inclua-as com o parâmetro `-Credential`. Por exemplo, o comando a seguir usa as credenciais da função e só funcionará se for concedida a permissão `iam:ListRoles` à função e o cmdlet `Get-IAMRoles` puder ser executado:

```
PS C:\> get-iamroles -Credential $Creds
```

4. Para retornar para as suas credenciais originais, basta parar de usar o parâmetro `-Credentials $Creds` e permitir que o PowerShell reverta para as credenciais que estão armazenadas no perfil padrão.

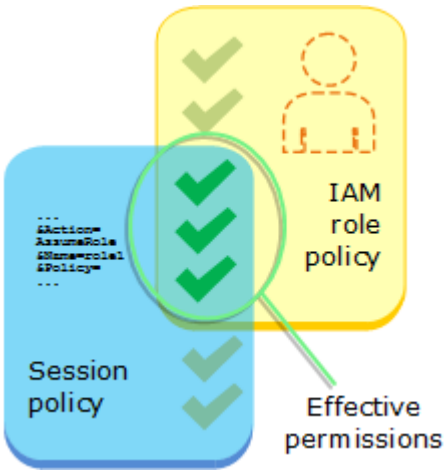
Alternar para um perfil do IAM (API da AWS)

Uma função especifica um conjunto de permissões que você pode usar para acessar os recursos da AWS. Nesse sentido, ela é semelhante a um usuário do IAM. Um principal (pessoa ou aplicativo) assume uma função para receber permissões temporárias para executar as tarefas necessárias e interagir com recursos do AWS. O perfil pode estar em sua própria conta ou em qualquer outra Conta da AWS. Para obter mais informações sobre funções, seus benefícios e como criar e configurá-las, consulte Perfis do IAM e Criar um perfil do IAM. Para saber mais sobre os diferentes métodos que você pode usar para assumir uma função, consulte Métodos para assumir um perfil.

> **Important**
>
> As permissões do usuário do IAM e quaisquer funções você venha a assumir não são cumulativas. Apenas um conjunto de permissões é ativo por vez. Quando você assume um perfil, perde temporariamente as permissões de usuário ou perfil anteriores e trabalha com as permissões atribuídas ao perfil. Ao sair da função, suas permissões originais serão, automaticamente, restauradas.

Para assumir uma função, um aplicativo chama a operação de API AWS STS`AssumeRole` do e transmite o ARN da função a ser usada. A operação cria uma nova sessão com credenciais temporárias. Esta sessão tem as mesmas permissões que as políticas baseadas em identidade para essa função.

Quando você chamar `AssumeRole`, poderá transmitir as políticas de políticas de sessão gerenciadas ou em linha. As políticas de sessão são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão de credencial temporária para um perfil ou sessão de usuário federado. É possível passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. É possível usar o parâmetro `PolicyArns` para especificar até 10 políticas de sessão gerenciadas. As permissões da sessão resultam da interseção das políticas baseadas em identidade da entidade e das políticas de sessão. As políticas de sessão são úteis quando você precisa fornecer as credenciais temporárias da função para outra pessoa. A outra pessoa pode usar as credenciais temporárias da função em chamadas subsequentes à API da AWS para acessar recursos na conta que possui a função. Você não pode usar políticas de sessão para conceder mais permissões do que as permitidas pela política baseada em identidade. Para saber mais sobre como a AWS determina as permissões efetivas de uma função, consulte Lógica da avaliação de política.



Você pode chamar `AssumeRole` quando tiver se conectado como um usuário do IAM ou como um usuário autenticado externamente (SAML ou OIDC) que já esteja usando um perfil. Você também pode usar o encadeamento de funções, que é o uso de uma função para assumir uma segunda função. Você não pode assumir um perfil quando está conectado como o Usuário raiz da conta da AWS.

Por padrão, a sessão do perfil dura uma hora. Quando assume esta função usando as operações de API AWS STS`AssumeRole* do` , você pode especificar uma valor para o parâmetro `DurationSeconds`. Esse valor pode variar de 900 segundos (15 minutos) até o valor configurado de duração máxima da sessão para o perfil. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil.

Se você usar o encadeamento de funções, a sessão será limitada a uma duração máxima de uma hora. Se você usar o parâmetro `DurationSeconds` para fornecer um valor maior do que uma hora, a operação falhará.

> **Note**
>
> Por motivos de segurança, os administradores podem revisar logs do AWS CloudTrail para saber quem executou uma ação na AWS. Seu administrador pode exigir que você especifique uma identidade-fonte ou um nome de sessão de função ao assumir a função. Para obter mais informações, consulte `sts:SourceIdentity` e `sts:RoleSessionName`.

Os exemplos de código a seguir mostram como criar um usuário e assumir um perfil.

> **Warning**
>
> Para evitar riscos de segurança, não use usuários do IAM para autenticação ao desenvolver software com propósito específico ou trabalhar com dados reais. Em vez disso, use federação com um provedor de identidade, como Centro de Identidade do AWS IAM.

- Crie um usuário sem permissões.
- Crie uma função que conceda permissão para listar os buckets do Amazon S3 para a conta.
- Adicione uma política para permitir que o usuário assuma a função.
- Assuma o perfil e liste buckets do S3 usando credenciais temporárias, depois limpe os recursos.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
```

```
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
```

```
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
```

```
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
```

```
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
```

```
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }
```

```
        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
```

```
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
```

```
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
 ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
 ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
accessKeyId });
```

```
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();
```

```
        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
```

```
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");
```

```
        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);
```

```
        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
patience.");
    }
```

```
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
```

```
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
```

```
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
```

```
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }
```

```
    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para .NET.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
```

```
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
```

```
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
```

```
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
```

```
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
  export AWS_SECRET_ACCESS_KEY=$key_secret

  local buckets
  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. This should not have
happened."
  else
    errecho "Because the role with permissions has not been assumed, listing
buckets failed."
  fi

  echo
  echo_repeat "*" 88
  echo "Now assume the role $iam_role_name and list the buckets."
  echo_repeat "*" 88
  echo

  local credentials

  credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
   echo "the assumed role."
 else
   errecho "Failed to list buckets. This should not happen."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   export AWS_SESSION_TOKEN=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 local result=0
 export AWS_ACCESS_KEY_ID=""
 export AWS_SECRET_ACCESS_KEY=""

 echo
 echo_repeat "*" 88
```

```
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

As funções do IAM usadas neste cenário.

```
###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}
```

```
  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
```

```
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user already exists, we don't want to try to create it.
if (iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name already exists in the account."
  return 1
fi

response=$(aws iam create-user --user-name "$user_name" \
  --output text \
  --query 'User.Arn')

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-user operation failed.$response"
  return 1
fi

echo "$response"
```

```
  return 0
}

################################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
```

```
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
```

```
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_policy"
  echo "Creates an AWS Identity and Access Management (IAM) policy."
  echo "  -n policy_name   The name of the IAM policy."
  echo "  -p policy_json -- The policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
```

```
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
```

```
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
```

```
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_delete_policy"
  echo "Deletes an AWS Identity and Access Management (IAM) policy"
  echo "  -n policy_arn -- The name of the IAM policy arn."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
```

```
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

########################################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
```

```
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
}

# Retrieve the calling parameters.
while getopts "u:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user does not exist, we don't want to try to delete it.
if (! iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name does not exist in the account."
  return 1
fi

response=$(aws iam delete-user \
  --user-name "$user_name")

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
```

```
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;
```

```
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
```

```
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }
```

```
        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
```

```
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());
```

```
        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
```

```
            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }
```

```
        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para C++.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/aws-sdk-go-v2/service/s3"
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
```

```
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}
```

```
// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
```

```
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
```

```
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)
```

```
 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
```

```
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
```

```
if scenario.questioner.AskBool(
 "Do you want to delete the resources created for this example? (y/n)", "y",
) {
 policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
*role.RoleName)
 if err != nil {
  panic(err)
 }
 for _, policy := range policies {
  err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
*policy.PolicyArn)
  if err != nil {
   panic(err)
  }
  err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
  if err != nil {
   panic(err)
  }
  log.Printf("Detached policy %v from role %v and deleted the policy.\n",
   *policy.PolicyName, *role.RoleName)
 }
 err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Deleted role %v.\n", *role.RoleName)

 userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 for _, userPol := range userPols {
  err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
 }
 keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 for _, key := range keys {
```

```
    err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
    if err != nil {
     panic(err)
    }
    log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
   }
   err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

Defina um struct que encapsule as ações de conta.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
```

```
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Defina um struct que encapsule as ações de política.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
```

```
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
```

```
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
```

```
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Defina um struct que encapsule as ações de perfil.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

```
// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
```

```
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
```

```
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
```

```
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
```

```
 return err
}
```

Defina um struct que encapsule as ações de usuário.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

```
// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

```
// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
```

```
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

```
// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Go.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações do usuário do IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
```

```
                    <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        String policyName = args[1];
        String roleName = args[2];
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
```

```
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
```

```
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();
```

```
            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();
```

```
            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);

            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
```

```
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário e um perfil do IAM que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função. Após assumir a função, use credenciais temporárias para listar os buckets para a conta.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
```

```
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
```

```
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
```

```
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
```

```
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});
```

```
// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);

// Clean up.
await iamClient.send(
  new DetachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeletePolicyCommand({
```

```
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para JavaScript.
  - AttachRolePolicy

- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações do usuário do IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
```

```
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }
```

```
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
```

```
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
```

```
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
```

```
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }
```

```
    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
```

```
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
```

```
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para PHP.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário e um perfil do IAM que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função. Após assumir a função, use credenciais temporárias para listar os buckets para a conta.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
```

```
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
```

```
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
```

```
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
"
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
    )
    try:
```

```
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
```

```
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
```

```
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - AttachRolePolicy

- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário e um perfil do IAM que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função. Após assumir a função, use credenciais temporárias para listar os buckets para a conta.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
```

```
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
```

```
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
```

```
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 'sts:AssumeRole',
        Resource: role.arn
      }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
```

```
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
     break if count.zero?
   end
 rescue Aws::Errors::ServiceError => e
   if e.code == 'AccessDenied'
     puts('Attempt to list buckets with no permissions: AccessDenied.')
   else
     @logger.info("Couldn't list buckets for the account. Here's why: ")
     @logger.info("\t#{e.code}: #{e.message}")
     raise
   end
 end

 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
```

```
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Deletes a user. If the user has inline policies or access keys, they are
deleted
 # before the user is deleted.
 #
 # @param user [Aws::IAM::User] The user to delete.
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
```

```
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
```

```
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Ruby.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
```

```
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
```

```
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
```

```
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
```

```
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Use um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2

As aplicações executadas na instância do Amazon EC2 devem incluir credenciais da AWS nas solicitações de API da AWS. Você poderia fazer com que os desenvolvedores armazenassem as credenciais da AWS diretamente na instância do Amazon EC2 e permitissem que as aplicações nessa instância usassem essas credenciais. Contudo, os desenvolvedores teriam que gerenciar as credenciais e garantir a transmissão segura das credenciais para cada instância e atualização de cada instância do Amazon EC2 na hora certa para fazer a atualização das credenciais. Isso é muito trabalho adicional.

Em vez disso, você pode e deve usar um perfil do IAM para gerenciar credenciais temporárias para aplicações executadas em uma instância do Amazon EC2. Quando você usa um perfil, não é necessário distribuir credenciais de longo prazo (como credenciais de login ou chaves de acesso) para uma instância do Amazon EC2. Em vez disso, a função fornece permissões temporárias que os aplicativos podem usar ao fazer chamadas para outros recursos da AWS. Quando você executa uma instância do Amazon EC2, você especifica uma função do IAM para associar à instância.

Os aplicativos que são executados na instância, por sua vez, usam as credenciais temporárias fornecidas pela função para assinar solicitações da API.

O uso de perfis para conceder permissões a aplicações que são executadas em instâncias do Amazon EC2 exige configuração adicional. Um aplicativo em execução em uma instância do Amazon EC2 é abstraído da AWS pelo sistema operacional virtualizado. Devido a essa separação adicional, é necessário ter uma etapa adicional para atribuir um perfil da AWS e suas permissões associadas a uma instância do Amazon EC2 e torná-las disponíveis para suas aplicações. Essa etapa adicional é a criação de um perfil da instância anexado à instância. O perfil da instância contém a função e pode fornecer as credenciais temporárias da função para um aplicativo que é executado na instância. Essas credenciais temporárias podem ser usadas em chamadas da API do aplicativo para acessar recursos e limitar o acesso apenas aos recursos que a função especifica.

> **Note**
>
> Só é possível atribuir um perfil a uma instância do Amazon EC2 por vez, e todas as aplicações na instância compartilham os mesmos perfil e permissões. Ao utilizar o Amazon ECS para gerenciar suas instâncias do Amazon EC2, é possível atribuir perfis às tarefas do Amazon ECS que podem ser diferenciadas do perfil da instância do Amazon EC2 na qual ela está sendo executada. A atribuição de um perfil a cada tarefa se alinha ao princípio do acesso de privilégio mínimo e permite um maior controle granular sobre ações e recursos. Para obter mais informações, consulte Uso de perfis do IAM com tarefas do Amazon ECS no Guia das práticas recomendadas do Amazon Elastic Container Service.

Usar funções dessa forma tem vários benefícios. Como as credenciais de perfil são temporárias e atualizadas automaticamente, não é necessário gerenciar credenciais, e você não precisa se preocupar com os riscos de segurança de longo prazo. Além disso, se usar uma única função para várias instâncias, você pode fazer uma alteração nessa função e a alteração se propagará automaticamente para todas as instâncias.

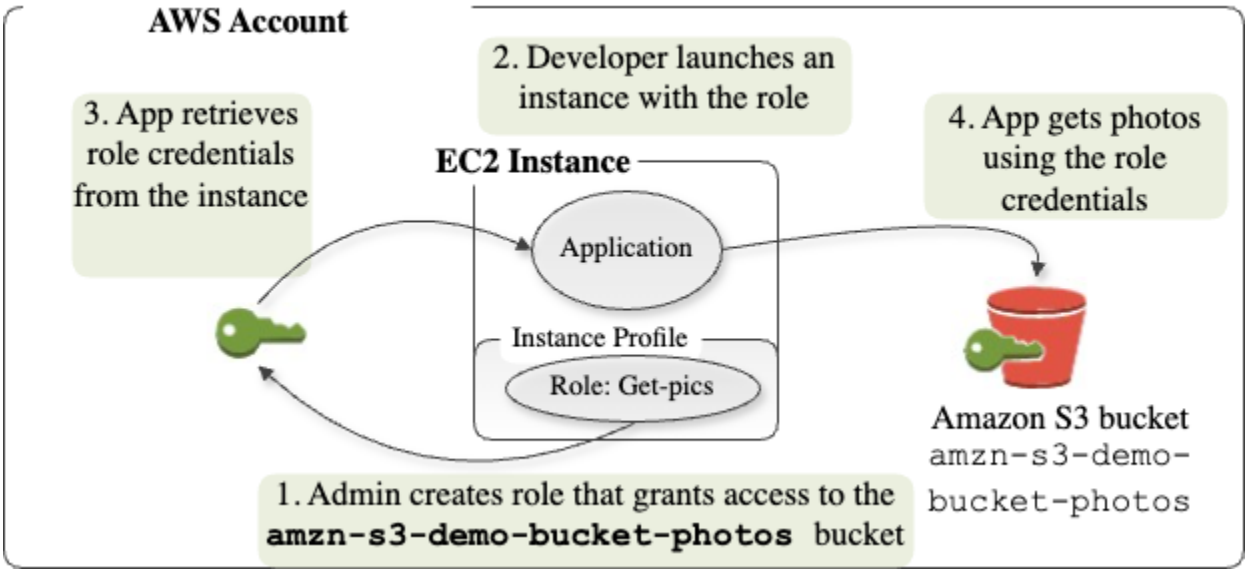
> **Note**
>
> Embora geralmente um perfil seja atribuído a uma instância do Amazon EC2 quando você a inicia, também é possível anexar um perfil a uma instância do Amazon EC2 que já esteja em execução. Para saber como anexar uma função a uma instância em execução, consulte Funções do IAM do Amazon EC2.

Tópicos

- Como os perfis para as instâncias do Amazon EC2 funcionam?
- Permissões necessárias para usar funções com o Amazon EC2
- Como faço para começar?
- Informações relacionadas

Como os perfis para as instâncias do Amazon EC2 funcionam?

Na figura a seguir, um desenvolvedor executa um aplicativo em uma instância do Amazon EC2 que requer acesso ao bucket do S3 denominado `amzn-s3-demo-bucket-photos`. Um administrador cria o perfil de serviço `Get-pics` e anexa o perfil à instância do Amazon EC2. A função inclui uma política de permissões que concede acesso somente leitura ao bucket do S3 especificado. Ela também inclui uma política de confiança que permite que a instância do Amazon EC2 assuma o perfil e recupere as credenciais temporárias. Quando o aplicativo é executado na instância, ele pode usar as credenciais temporárias da função para acessar o bucket de fotos. O administrador não precisa conceder ao desenvolvedor permissão para acessar o bucket de fotos, e o desenvolvedor nunca precisa compartilhar nem gerenciar as credenciais.



1. O administrador usa o IAM para criar a função **`Get-pics`**. Na política de confiança do perfil, o administrador especifica que apenas as instâncias do Amazon EC2 podem assumir o perfil. Na política de permissões da função, o administrador especifica a permissão somente leitura para o bucket de `amzn-s3-demo-bucket-photos`.

2. Um desenvolvedor inicia uma instância do Amazon EC2 e atribui o perfil `Get-pics` a essa instância.

   > **Note**
   >
   > Se você usar o console do IAM, o perfil da instância será tanto gerenciado quanto mais transparente para você. No entanto, se você usar a AWS CLI ou a API para criar e gerenciar o perfil e a instância do Amazon EC2, será necessário criar o perfil de instância e atribuir o perfil a ela em etapas distintas. Assim, quando você executar a instância, será necessário especificar o nome de perfil da instância em vez do nome da função.

3. Quando a aplicação é executada, ela obtém as credenciais de segurança temporárias dos metadados da instância do Amazon EC2, conforme descrito em Recuperar credenciais de segurança de metadados da instância. Essas são as credenciais de segurança temporárias que representam a função e são válidas por um período limitado.

   Com alguns SDKs da AWS, o desenvolvedor pode usar um provedor que gerencia as credenciais de segurança temporárias de forma transparente. (A documentação para SDKs individuais da AWS descreve os recursos suportados pelo SDK para o gerenciamento de credenciais).

   Como alternativa, o aplicativo pode obter as credenciais temporárias diretamente dos metadados da instância do Amazon EC2. Credenciais e valores relacionados estão disponíveis na `iam/security-credentials/`*`role-name`* categoria (nesse caso, `iam/security-credentials/Get-pics`) dos metadados. Se o aplicativo obtiver as credenciais dos metadados da instância, ele poderá armazenar as credenciais em cache.

4. Ao usar as credenciais temporárias recuperadas, o aplicativo acessa o bucket de fotos. Devido a política anexada à função **`Get-pics`**, o aplicativo tem permissões somente leitura.

   As credenciais temporárias de segurança disponíveis na instância são atualizadas automaticamente antes de perder a validade. Assim, há sempre um conjunto válido disponível. O aplicativo só precisa se certificar de receber um novo conjunto de credenciais dos metadados da instância antes que as atuais expirem. É possível usar o SDK da AWS para gerenciar credenciais para que a aplicação não precise incluir uma lógica adicional para atualizar as credenciais. Por exemplo, instanciando clientes com provedores de credenciais de perfis da instância. No entanto, se o aplicativo receber credenciais de segurança temporárias dos metadados da instância e os tiver armazenados em cache, ele deverá obter um conjunto de credenciais atualizadas a cada hora, ou pelo menos 15 minutos antes de o conjunto atual expirar. O tempo

de validade está incluído nas informações que são retornadas na categoria `iam/security-credentials/`*`role-name`*.

Permissões necessárias para usar funções com o Amazon EC2

Para iniciar uma instância com um perfil, o desenvolvedor deve ter permissão para iniciar instâncias do Amazon EC2 e para transmitir perfis do IAM.

A política de exemplo a seguir permite que os usuários usem o Console de gerenciamento da AWS para executar uma instância com uma função. A política inclui curingas (*) para permitir que um usuário transmita qualquer função e execute as ações listadas do Amazon EC2. A ação `ListInstanceProfiles` permite que os usuários visualizem todas as perfis disponíveis na Conta da AWS.

Example Exemplo de política que concede a um usuário permissão para usar o console do Amazon EC2 para executar uma instância com qualquer função

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Restringir quais perfis podem ser transmitidos para as instâncias do Amazon EC2 (usando PassRole)

Você pode usar a permissão `PassRole` para restringir quais perfis um usuário pode transmitir a uma instância do Amazon EC2 quando o usuário inicia a instância. Isso ajuda a evitar que o usuário execute aplicações que tenham mais permissões do que o usuário recebeu, ou seja, de poder obter privilégios elevados. Por exemplo, imagine que a usuária Alice tenha permissões apenas para iniciar instâncias do Amazon EC2 e trabalhar com buckets do Amazon S3, mas o perfil que ela transmite para uma instância do Amazon EC2 tem permissões para trabalhar com o IAM e o Amazon DynamoDB. Nesse caso, Alice pode ser capaz de executar a instância, fazer login nela, obter credenciais de segurança temporárias e, em seguida, executar ações no IAM ou no DynamoDB para as quais ela não tem autorização.

Para restringir quais perfis um usuário pode transmitir para uma instância do Amazon EC2, você cria uma política que permita a ação `PassRole`. Em seguida, anexe a política ao usuário (ou a um grupo do IAM ao qual o usuário pertença) que iniciará as instâncias do Amazon EC2. No elemento `Resource` da política, liste o perfil ou perfis que o usuário tem permissão para transmitir para instâncias do Amazon EC2. Quando o usuário executa uma instância e associa uma função a ela, o Amazon EC2 verifica se o usuário tem permissão para transmitir essa função. Naturalmente, você também deve garantir que a função que o usuário pode transmitir não inclua mais permissões do que o usuário deve ter.

> **Note**
>
> `PassRole` não é uma ação da API da forma que `RunInstances` ou `ListInstanceProfiles` é. Em vez disso, é uma permissão que a AWS verifica sempre que um ARN da função é transmitido como um parâmetro para uma API (ou o console faz isso em nome do usuário). Ele ajuda um administrador a controlar quais funções podem ser transmitidas por quais usuários. Nesse caso, ele garante que o usuário tenha permissão para anexar uma função específica a uma instância do Amazon EC2.

Example Exemplo de política que concede permissão para um usuário iniciar uma instância do Amazon EC2 com um perfil específico

A política de exemplo a seguir permite que os usuários usem a API do Amazon EC2 para executar uma instância com uma função. O elemento `Resource` especifica o nome de recurso da Amazon (ARN) de uma função. Ao especificar o ARN, a política concede ao usuário a permissão para transmitir apenas a função `Get-pics`. Se o usuário tentar especificar uma função diferente ao executar uma instância, a ação falhará. O usuário não tem permissões para executar qualquer instância, independentemente de ele transmitir uma função.

JSON

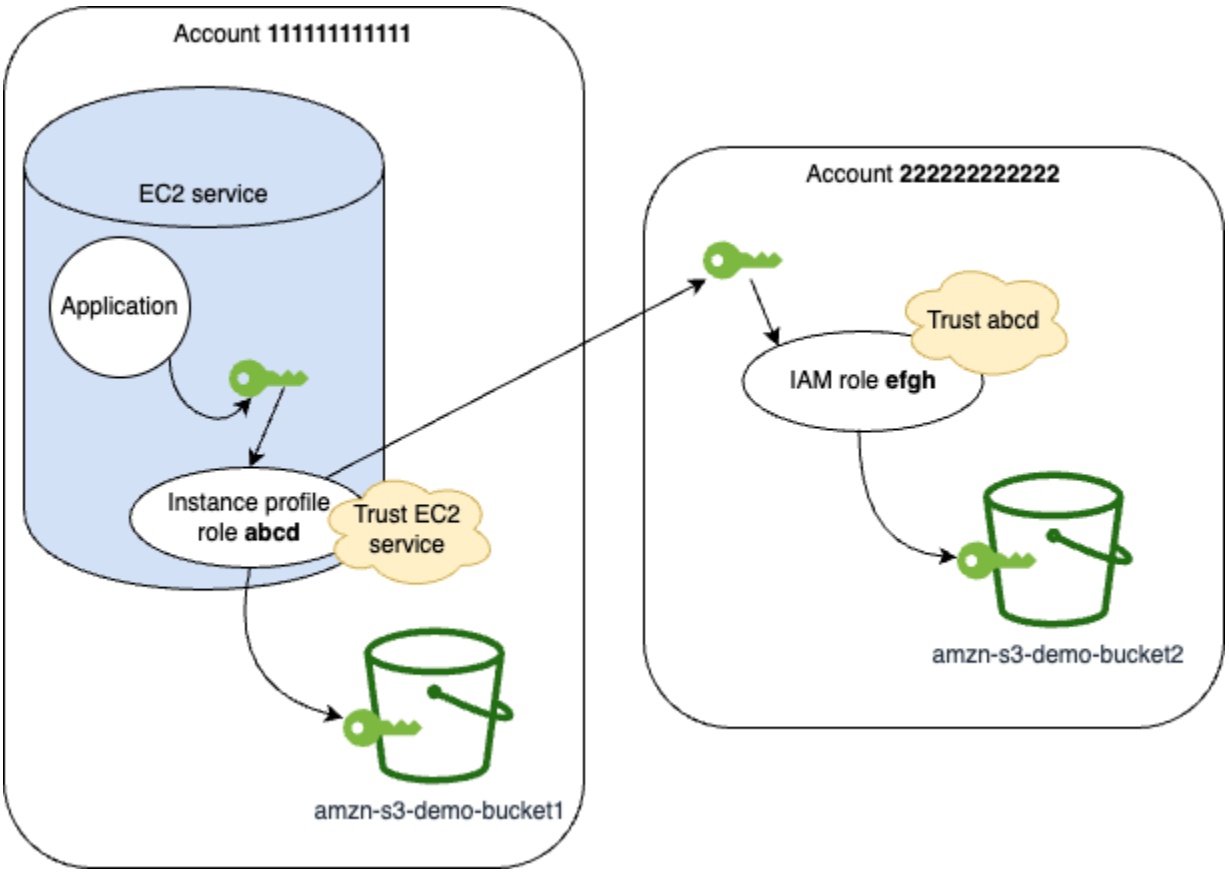
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/Get-pics"
        }
    ]
}
```

Permitir que uma função de perfil de instância alterne para uma função em outra conta

Você pode permitir que uma aplicação em execução em uma instância do Amazon EC2 execute comandos em outra conta. Para fazer isso, é necessário permitir que o perfil de instância do Amazon EC2 na primeira conta alterne para um perfil na segunda conta.

Imagine que você esteja usando duas Contas da AWS e queira permitir que uma aplicação em execução em uma instância do Amazon EC2 execute comandos da AWS CLI nas duas contas. Vamos supor que a instância do Amazon EC2 exista na conta `111111111111`. Essa instância inclui a função de perfil da instância `abcd` que permite que a aplicação execute tarefas somente leitura do Amazon S3 no bucket `amzn-s3-demo-bucket1` dentro da mesma conta `111111111111`. No

entanto, a aplicação também deve ter permissão para assumir a função `efgh` entre contas para acessar o bucket `amzn-s3-demo-bucket2` do Amazon S3 na conta `222222222222`.



A função de perfil de instância `abcd` do Amazon EC2 deve ter a seguinte política de permissões para permitir que o aplicativo acesse o bucket `amzn-s3-demo-bucket1` do Amazon S3:

Política de permissões da função ***abcd*** da conta 111111111111

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
```

```
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1/*",
                "arn:aws:s3:::amzn-s3-demo-bucket1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

A função `abcd` deve confiar no serviço do Amazon EC2 para assumir a função. Para fazer isso, a função `abcd` deve ter a seguinte política de confiança:

Política de confiança da função ***abcd*** da conta 111111111111

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
```

```
}
```

Vamos supor que a função entre contas `efgh` permita tarefas somente leitura do Amazon S3 no bucket `amzn-s3-demo-bucket2` na mesma conta `222222222222`. Para fazer isso, a função entre contas `efgh` deve ter a seguinte política de permissões:

Política de permissões da função ***efgh*** da conta 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket2/*",
                "arn:aws:s3:::amzn-s3-demo-bucket2"
            ]
        }
    ]
}
```

A função `efgh` deve confiar na função de perfil de instância `abcd` para assumi-la. Para fazer isso, a função `efgh` deve ter a seguinte política de confiança:

Política de confiança da função ***efgh*** da conta 222222222222

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Como faço para começar?

Para entender como os perfis funcionam com as instâncias do Amazon EC2, é necessário usar o console do IAM para criar um perfil, iniciar uma instância do EC2 que use esse perfil e, em seguida, examinar a instância em execução. Examine os metadados da instância para ver como as credenciais temporárias da função são disponibilizadas para uma instância. Você também pode ver como um aplicativo que é executado em uma instância pode usar a função. Use os recursos a seguir para saber mais.

-
- Demonstrações do SDK. A documentação do AWS SDK inclui demonstrações que mostram um aplicativo em execução em uma instância do Amazon EC2 que usa credenciais temporárias para perfis que fazem a leitura de um bucket do Amazon S3. Cada uma das demonstrações a seguir apresenta etapas semelhantes com uma linguagem de programação diferente:
  - Configurar funções do IAM para o Amazon EC2 com o SDK for Java no Guia do desenvolvedor do AWS SDK para Java
  - Launch an Amazon EC2 Instance using the SDK for .NET (Iniciar uma instância do Amazon EC2 usando o SDK para .NET) no Guia do desenvolvedor do AWS SDK para .NET

- Criar uma instância do Amazon EC2 com o SDK for Ruby no Guia do desenvolvedor do AWS SDK para Ruby

Informações relacionadas

Para obter mais informações sobre a criação de perfis ou funções para instâncias do Amazon EC2, consulte as seguintes informações:

- Para obter mais informações sobre como usar perfis do IAM com instâncias do Amazon EC2, consulte o Guia do usuário do Amazon EC2.
- Para criar uma função, consulte Criar um perfil do IAM
- Para obter mais informações sobre o uso de credenciais de segurança temporárias, consulte Credenciais de segurança temporárias no IAM.
- Se você trabalha com a API do IAM ou a CLI, é necessário criar e gerenciar os perfis de instância do IAM. Para obter mais informações sobre os perfis da instância, consulte Usar perfis de instância.
- Para obter mais informações sobre credenciais temporárias de segurança para perfis nos metadados da instância, consulte Recuperar credenciais de segurança dos metadados da instância no Guia do usuário do Amazon EC2.

Usar perfis de instância

Use um perfil de instância para passar uma função do IAM para uma instância do EC2. Para obter mais informações, consulte Perfis do IAM para o Amazon EC2 no Guia do usuário do Amazon EC2.

Gerenciar perfis de instância (console)

Se você usar o Console de gerenciamento da AWS para criar uma função para o Amazon EC2, o console criará automaticamente um perfil de instância e dará a ele o mesmo nome da função. Ao usar o console do Amazon EC2 para executar uma instância com uma função do IAM, você pode selecionar uma função para associar à instância. No console, a lista exibida é, na verdade, uma lista de nomes de perfis de instância. O console não cria um perfil de instância para uma função que não esteja associada ao Amazon EC2.

Você pode usar o Console de gerenciamento da AWS para excluir funções do IAM e perfis de instância do Amazon EC2 se a função e o perfil da instância tiverem o mesmo nome. Para saber mais sobre como excluir perfis de instância, consulte Excluir perfis ou perfis de instância.

> **Note**
>
> Para atualizar as permissões de uma instância, substitua o perfil da instância. Não é recomendável remover um perfil de um perfil de instância, pois existe um atraso de até uma hora antes que essa alteração entre em vigor.

Gerenciar perfis de instância (AWS CLI ou API da AWS)

Se gerenciar suas funções da AWS CLI ou da API da AWS, você criará funções e perfis de instância como ações separadas. Como funções e perfis de instância podem ter nomes diferentes, você precisa saber os nomes de seus perfis de instância, bem como os nomes das funções que eles contêm. Dessa forma, você pode escolher o perfil de instância correto ao executar uma instância do EC2.

Você pode anexar etiquetas aos seus recursos do IAM, incluindo perfis de instância, para identificar, organizar e controlar o acesso a eles. Os perfis de instância só podem ser marcados quando você usa a AWS CLI ou a API da AWS.

> **Note**
>
> Um perfil de instância pode conter apenas uma função do IAM, embora uma função possa ser incluída em vários perfis de instância. Esse limite de uma função por perfil de instância não pode ser aumentado. Você pode remover a função existente e, em seguida, adicionar uma função diferente a um perfil de instância. Você deve então esperar que a mudança apareça em toda a AWS devido à eventual consistency. Para forçar a alteração, desassocie o perfil de instância, associe o perfil de instância, ou interrompa a instância e, em seguida, reinicie-a.

Gerenciar perfis de instância (AWS CLI)

Você pode usar os seguintes comandos da AWS CLI para trabalhar com perfis de instância em uma conta da AWS.

- Criar um perfil de instância: `aws iam create-instance-profile`
- Marcar um perfil de instância: `aws iam tag-instance-profile`
- Listar tags para um perfil de instância: `aws iam list-instance-profile-tags`

- Desmarcar um perfil de instância: `aws iam untag-instance-profile`
- Adicionar uma função a um perfil de instância: `aws iam add-role-to-instance-profile`
- Listar perfis de instância: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Obter informações sobre um perfil de instância: `aws iam get-instance-profile`
- Remover uma função de um perfil de instância: `aws iam remove-role-from-instance-profile`
- Excluir um perfil de instância: `aws iam delete-instance-profile`

Você também pode anexar uma função a uma instância do EC2 já em execução usando os seguintes comandos. Para obter mais informações, consulte Funções do IAM para o Amazon EC2.

- Anexar um perfil de instância com uma função a uma instância do EC2 em execução ou interrompida: `aws ec2 associate-iam-instance-profile`
- Obter informações sobre um perfil de instância anexado a uma instância do EC: `aws ec2 describe-iam-instance-profile-associations`
- Desanexar um perfil de instância com uma função de uma instância do EC2 em execução ou interrompida: `aws ec2 disassociate-iam-instance-profile`

Gerenciar perfis de instância (API da AWS)

Você pode chamar as seguintes operações de API da AWS para trabalhar com perfis de instância em uma Conta da AWS.

- Criar um perfil de instância: `CreateInstanceProfile`
- Marcar um perfil de instância: `TagInstanceProfile`
- Listar tags em um perfil de instância: `ListInstanceProfileTags`
- Desmarcar um perfil de instância: `UntagInstanceProfile`
- Adicionar uma função a um perfil de instância: `AddRoleToInstanceProfile`
- Listar perfis de instância: `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Obter informações sobre um perfil de instância: `GetInstanceProfile`
- Remover uma função de um perfil de instância: `RemoveRoleFromInstanceProfile`
- Excluir um perfil de instância: `DeleteInstanceProfile`

Você também pode anexar uma função a uma instância do EC2 já em execução chamando as seguintes operações. Para obter mais informações, consulte Funções do IAM para o Amazon EC2.

- Anexar um perfil de instância com uma função a uma instância do EC2 em execução ou interrompida: `AssociateIamInstanceProfile`
- Obter informações sobre um perfil de instância anexado a uma instância do EC: `DescribeIamInstanceProfileAssociations`
- Desanexar um perfil de instância com uma função de uma instância do EC2 em execução ou interrompida: `DisassociateIamInstanceProfile`

Federação e provedores de identidade na AWS

Como uma prática recomendada, sugerimos que você exija que os usuários humanos usem uma federação com um provedor de identidade para acessar recursos da AWS, em vez de criar usuários do IAM individuais na Conta da AWS deles. Com um provedor de identidade (IdP), você pode gerenciar suas identidades de usuários fora da AWS e fornecer a esses usuários externos permissões para usar recursos da AWS na sua conta. Isso será útil se a sua organização já tiver seu próprio sistema de identidade, como um diretório de usuários corporativos. Também será útil se você criar um aplicativo móvel ou web que precise de acesso aos recursos da AWS.

> **Note**
>
> Você também pode gerenciar usuários humanos no Centro de Identidade do IAM com um provedor de identidade do SAML externo em vez de usar a federação do SAML no IAM. A federação do Centro de Identidade do IAM com um provedor de identidade oferece a capacidade de conceder acesso a várias contas da AWS em sua organização e a várias aplicações da AWS. Para obter informações sobre situações específicas em que um usuário do IAM é necessário, consulte Quando criar um usuário do IAM (em vez de um perfil). Você também pode usar o gerenciador de acesso à conta — um atributo do IAM que permite atribuir perfis do IAM existentes nas contas da sua organização aos usuários e grupos do IAM Identity Center. Você pode usá-lo junto com os conjuntos de permissões ou sozinho.

Caso prefira usar uma única conta da AWS sem habilitar o Centro de Identidade do IAM, você pode usar o IAM com um IdP externo que fornece informações de identidade à AWS usando o OpenID Connect (OIDC) ou o SAML 2.0 (Security Assertion Markup Language 2.0). O OIDC conecta aplicações, como o GitHub Actions, que não são executados em recursos da AWS, a recursos

da AWS Exemplos de provedores de identidade SAML bem conhecidos são Shibboleth e Active Directory Federation Services.

Quando você usa um provedor de identidade do , não precisa criar código de login personalizado nem gerenciar suas próprias identidades de usuários. O IdP faz isso para você. Os usuários externos fazem login por meio do IdP, e você pode conceder a essas identidades externas permissões para usar os recursos da AWS na sua conta. Os provedores de identidade ajudam a manter sua Conta da AWS segura, pois você não precisa distribuir ou incorporar credenciais de segurança de longo prazo, como chaves de acesso.

Revise a tabela a seguir para ajudar a determinar qual é o melhor tipo de federação do IAM para seu caso de uso: IAM, Centro de Identidade do IAM ou Amazon Cognito. Os resumos e a tabela a seguir fornecem uma visão geral dos métodos que os usuários podem empregar para ter acesso federado aos recursos da AWS.

Tipo de Federação do IAM	Account type (Tipo de conta)	Gerenciamento de acesso de...	Fonte de identidades compatível
Federação com o IAM Identity Center	Várias contas gerenciadas pelo AWS Organizations	Os usuários humanos da sua força de trabalho	• SAML 2.0 • Active Directory gerenciado • Diretório do Identity Center
Federação com o IAM	Conta autônoma única	• Usuários humanos em implantações de curto prazo e pequena escala • Usuários que são máquinas	• SAML 2.0 • OIDC
Federação com bancos de identidades do Amazon Cognito	Any	Os usuários de aplicações que requerem autorização do IAM para acessar os recursos	• SAML 2.0 • OIDC • Selecionar provedores de identidades sociais do OAuth 2.0

Federação com o IAM Identity Center

Para gerenciamento de acesso centralizado de seres humanos, recomendamos que você use o IAM Identity Center para gerenciar o acesso às suas contas e as permissões dentro dessas contas. Os usuários do IAM Identity Center recebem credenciais de curto prazo para usar seus recursos da AWS. Você pode usar o Active Directory, um provedor de identidades (IdP) externo ou um diretório do IAM Identity Center como a fonte de identidades de usuários e grupos para conceder acesso aos seus recursos da AWS.

O IAM Identity Center é compatível com federação de identidades com SAML (Security Assertion Markup Language) 2.0 para fornecer acesso de login único federado aos usuários autorizados a usar as aplicações no portal de acesso da AWS. Assim, os usuários podem usar a autenticação única para entrar nos serviços compatíveis com SAML, inclusive o Console de gerenciamento da AWS e aplicações de terceiros, como o Microsoft 365, o SAP Concur e o Salesforce.

Federação com o IAM

Embora seja altamente recomendável gerenciar usuários humanos no IAM Identity Center, você pode habilitar o acesso de entidades principais federadas com o IAM para usuários humanos em implantações de curto prazo e pequena escala. O IAM permite que você use IdPs SAML 2.0 e Open ID Connect (OIDC) separados e atributos de entidades principais federadas para controle de acesso. Com o IAM, você pode passar os atributos do usuário, como centro de custos, título ou nacionalidade, dos IdPs para a AWS, e implementar permissões de acesso refinadas com base nesses atributos.

Uma workload é uma coleção de códigos e recursos que fornece valor comercial, como uma aplicação ou um processo de backend. Sua workload pode exigir uma identidade do IAM para fazer solicitações aos serviços, aplicações, ferramentas operacionais e componentes da AWS. Essas identidades incluem máquinas em execução em seus ambientes da AWS, como instâncias do Amazon EC2 ou funções do AWS Lambda.

Você também pode gerenciar identidades de máquina para partes externas que precisam de acesso. Para dar acesso a identidades de máquina, você pode usar perfis do IAM. Os perfis do IAM têm permissões específicas e fornecem uma maneira de acessar a AWS com base em credenciais de segurança temporárias com uma sessão de perfil. Além disso, você pode ter máquinas fora da AWS que precisam de acesso aos seus ambientes da AWS. Para máquinas que são executadas fora da AWS, você pode usar o IAM Roles Anywhere. Para ter mais informações sobre funções, consulte Perfis do IAM. Para obter detalhes sobre como usar perfis para delegar acesso em Contas da AWS, consulte Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM.

Para usar um IdP diretamente no IAM, você cria uma entidade provedora de identidades para estabelecer uma relação de confiança entre sua Conta da AWS e esse IdP. O IAM oferece suporte a IdPs compatíveis com OpenID Connect (OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Para obter mais informações sobre como usar um desses IdPs com a AWS, consulte as seguintes seções:

- Federação OIDC
- Federação SAML 2.0

Federação com bancos de identidades do Amazon Cognito

O Amazon Cognito destina-se a desenvolvedores que desejam autenticar e autorizar usuários em aplicações móveis e aplicações da Web. Os grupos de usuários do Amazon Cognito adicionam recursos de login e inscrição à aplicação, e os bancos de identidades fornecem as credenciais do IAM que concedem aos usuários acesso aos recursos protegidos que você gerencia na AWS. Os bancos de identidades obtêm credenciais para sessões temporárias por meio da operação da API `AssumeRoleWithWebIdentity`.

O Amazon Cognito trabalha com provedores de identidades externos compatíveis com SAML e OpenID Connect, e com provedores de identidades sociais, como o Facebook, o Google e a Amazon. A aplicação pode inscrever um usuário em um grupo de usuários ou um IdP externo e depois recuperar recursos em nome dele com sessões personalizadas temporárias em um perfil do IAM.

Recursos adicionais

- Para obter uma demonstração de como criar um proxy de federação personalizado que permita autenticação única (SSO) no Console de gerenciamento da AWS usando o sistema de autenticação da sua organização, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.
- Para obter instruções sobre como permitir que suas workloads da AWS acessem serviços externos sem armazenar credenciais, consulte Federação de identidades da AWS para serviços externos.

Cenários comuns

> **Note**
>
> Recomendamos exigir que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Você já pensou em usar o Centro de Identidade do AWS IAM? O IAM Identity Center pode ser usado para gerenciar centralmente o acesso a várias Contas da AWS e fornecer aos usuários acesso de logon único protegido por MFA a todas as contas atribuídas em um só lugar. Com o IAM Identity Center, é possível criar e gerenciar identidades de usuários no IAM Identity Center ou conectar facilmente ao provedor de identidades compatível com SAML 2.0 existente. Para obter mais informações, consulte O que é o IAM Identity Center? no Guia do usuário do Centro de Identidade do AWS IAM.

Você pode usar um provedor de identidades (IdP) externo para gerenciar identidades de usuários fora da AWS e do IdP externo. Um IdP externo pode fornecer informações de identidade à AWS usando OpenID Connect (OIDC) ou Security Assertion Markup Language (SAML). O OIDC é comumente usado quando uma aplicação que não é executada na AWS precisa de acesso a recursos da AWS.

Quando você desejar configurar uma federação com um IdP externo, crie um provedor de identidades do IAM para informar a AWS sobre o IdP externo e sua configuração. Isso estabelece confiança entre sua Conta da AWS e o IdP externo. Os tópicos a seguir fornecem cenários comuns para uso de provedores de identidade do IAM.

Tópicos

- Amazon Cognito para aplicativos móveis
- federação OIDC para aplicativos móveis

Amazon Cognito para aplicativos móveis

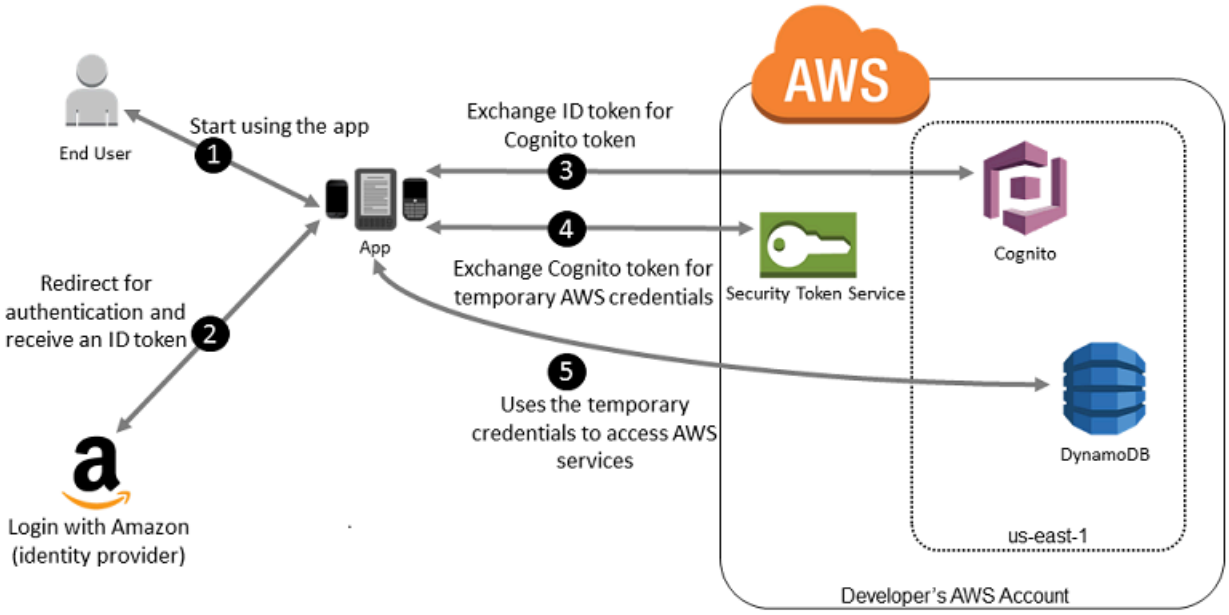
A maneira preferencial de usar a federação OIDC é usar o Amazon Cognito. Por exemplo, a desenvolvedora Adele está criando um jogo para um dispositivo móvel no qual os dados do usuário, como pontuações e perfis, são armazenados no Amazon S3 e no Amazon DynamoDB. Adele também pode armazenar esses dados localmente no dispositivo e usar o Amazon Cognito para mantê-los sincronizados entre os dispositivos. Ela sabe que por motivos de segurança e manutenção, as credenciais de segurança da AWS de longo prazo não devem ser distribuídas

com o jogo. Ela também sabe que o jogo pode ter um grande número de usuários. Por todos esses motivos, ela não quer criar novas identidades de usuários no IAM para cada jogador. Em vez disso, ela desenvolve o jogo de modo que os usuários possam fazer login usando uma identidade já estabelecida com um provedor de identidades (IdP) externo bem conhecido, como Login with Amazon, Facebook, Google ou qualquer IdP compatível com OpenID Connect (OIDC). O jogo dela pode aproveitar o mecanismo de autenticação de um desses provedores para validar a identidade do usuário.

Para permitir que o aplicativo móvel acesse seus recursos da AWS, Adele primeiro se registra para um ID de desenvolvedor com seus IdPs escolhidos. Ela também configura o aplicativo com cada um desses provedores. Na sua Conta da AWS que contém o bucket do Amazon S3 e a tabela do DynamoDB para o jogo, Adele usa o Amazon Cognito para criar perfis do IAM que definem com precisão as permissões necessárias para o jogo. Se ela estiver usando um IdP de OIDC, ela também criará uma entidade de provedor de identidades OIDC do IAM para estabelecer a confiança entre um grupo de identidades do Amazon Cognito na Conta da AWS e o IdP.

No código do aplicativo, a Adele chama a interface de login para o IdP que ela configurou anteriormente. O IdP lida com todos os detalhes para permitir que o usuário faça login, e o aplicativo obtenha um token de acesso OAuth ou um token de ID do OIDC do provedor. O aplicativo da Adele pode trocar essas informações de autenticação para um conjunto de credenciais de segurança temporárias que consiste em um ID de chave de acesso da AWS, uma chave de acesso secreta e um token de sessão. O aplicativo pode então usar essas credenciais para acessar serviços da web oferecidos pela AWS. O aplicativo é limitado às permissões que são definidas na função que ele assume.

A figura a seguir mostra um fluxo simplificado de como isso funciona usando o Login with Amazon como IdP. Para a etapa 2, o aplicativo também pode usar o Facebook, Google ou qualquer IdP compatível com o OIDC, mas isto não é mostrado aqui.



1. Um cliente inicia o aplicativo em um dispositivo móvel. O aplicativo solicita que o usuário faça login.
2. A aplicação usa os recursos do Login with Amazon para aceitar as credenciais do usuário
3. A aplicação usa operações de API do Amazon Cognito `GetId` e `GetCredentialsForIdentity` para trocar o token de ID do login com a Amazon por um token do Amazon Cognito. O Amazon Cognito, que foi configurado para confiar no seu projeto de login com a Amazon, gera um token que ele troca por credenciais de sessão temporárias com o AWS STS.
4. A aplicação recebe credenciais de segurança temporárias do Amazon Cognito. A aplicação também pode usar o fluxo de trabalho básico (clássico) no Amazon Cognito para recuperar tokens do AWS STS usando `AssumeRoleWithWebIdentity`. Para obter mais informações consulte Identity pools (federated identities) authentication flow (Fluxo de autenticação de grupos de identidades [identidades federadas] do Amazon Cognito) no Amazon Cognito Developer Guide (Guia do desenvolvedor do Amazon Cognito).
5. As credenciais de segurança temporárias podem ser usadas pelo aplicativo para acessar qualquer recurso da AWS exigido pelo aplicativo para operar. A função associada às credenciais de segurança temporárias e às políticas atribuídas determina o que pode ser acessado.

Use o processo a seguir para configurar sua aplicação para usar o Amazon Cognito para autenticar usuários e fornecer à aplicação acesso aos recursos da AWS. Para obter etapas específicas para realizar esse cenário, consulte a documentação do Amazon Cognito.

1. (Opcional) Cadastre-se como um desenvolvedor com Login with Amazon, Facebook, Google ou qualquer outro IdP compatível com OpenID Connect (OIDC) e configure uma ou mais aplicações com o provedor. Esta etapa é opcional, pois o Amazon Cognito também oferece suporte ao acesso não autenticado (de convidado) para seus usuários.
2. Acesse Amazon Cognito no Console de gerenciamento da AWS. Use o assistente do Amazon Cognito para criar um grupo de identidades, que é um contêiner que o Amazon Cognito usa para manter as identidades dos usuários finais organizadas para suas aplicações. Você pode compartilhar grupos de identidades entre aplicativos. Ao configurar um grupo de identidades, o Amazon Cognito cria uma ou duas funções do IAM (uma para identidades autenticadas e outra para identidades de “convidados” não autenticados) que definem permissões para os usuários do Amazon Cognito.
3. Integre o AWSAmplify com a aplicação e importe os arquivos necessários para usar o Amazon Cognito.
4. Crie uma instância do provedor de credenciais do Amazon Cognito, passando o ID do grupo de identidades, o número da sua Conta da AWS e o nome do recurso da Amazon (ARN) dos perfis que você associou ao grupo de identidades. O assistente do Amazon Cognito no Console de gerenciamento da AWS fornece o código de exemplo para ajudar você a começar.
5. Quando o seu aplicativo acessa um recurso da AWS, transmite as instâncias do provedor de credenciais para o objeto do cliente, que, por sua vez, transmite as credenciais de segurança temporárias ao cliente. As permissões para as credenciais são baseadas na função ou funções que você definiu anteriormente.

Para saber mais, consulte:

- Faça login (Android) na documentação da estrutura do AWS Amplify.
- Faça login (iOS) na documentação da estrutura do AWS Amplify.

federação OIDC para aplicativos móveis

Para obter os melhores resultados, use o Amazon Cognito como intermediador de identidades para quase todos os cenários de federação OIDC. O Amazon Cognito é fácil de usar e fornece recursos adicionais como acesso anônimo (não autenticado) e sincronização de dados do usuário entre dispositivos e provedores. No entanto, se você já criou uma aplicação que usa federação OIDC e chama manualmente a API `AssumeRoleWithWebIdentity`, continue a usá-la, pois a aplicação continuará funcionando corretamente.

O processo de uso de federação OIDC sem o Amazon Cognito segue esta visão geral:

1. Cadastre-se como desenvolvedor com o provedor de identidade externa (IdP) e configure seu aplicativo com o IdP, que fornece a você um ID exclusivo para seu aplicativo. (IdPs diferentes usam terminologia diferente para este processo. Esta descrição usa o termo configurar para o processo de identificação de sua aplicação com o IdP.) Cada IdP fornece a você um ID de aplicativo exclusivo para esse IdP, portanto, se você configurar o mesmo aplicativo com vários IdPs, seu aplicativo terá vários IDs. Você pode configurar vários aplicativos com cada fornecedor.

   Os links externos a seguir fornecem informações sobre como usar alguns dos provedores de identidade (IdPs) mais usados:

   - Centro do desenvolvedor do Login with Amazon
   - Adicione o login do Facebook ao seu aplicativo ou site no site de desenvolvedores do Facebook.
   - Uso de OAuth 2.0 para login (OpenID Connect) no site de desenvolvedores do Google.

   > **Important**
   >
   > Se você usar um provedor de identidade OIDC do Google, Facebook ou Amazon Cognito, não crie um provedor de identidade do IAM separado no Console de gerenciamento da AWS. A AWS tem esses provedores de identidade OIDC incorporados e disponíveis para seu uso. Ignore a etapa a seguir e vá diretamente para a criação de novas funções usando seu provedor de identidade.

2. Se você usar um IdP diferente do Google, Facebook ou Amazon Cognito compatível com OIDC, crie uma entidade de provedor de identidade do IAM para ele.
3. No IAM, crie uma ou mais funções. Para cada função, defina quem pode assumir a função (a política de confiança) e quais permissões os usuários da aplicação têm (a política de permissões). Geralmente, você cria uma função para cada IdP ao qual um aplicativo dá suporte. Por exemplo, você pode criar uma função assumida por uma aplicação se o usuário fizer login por meio do Login with Amazon, uma segunda função para a mesma aplicação se o usuário fizer login por meio do Facebook e uma terceira função para a aplicação se o usuário fizer login por meio do Google. Para o relacionamento de confiança, especifique o IdP (como a Amazon.com) como o `Principal` (a entidade confiável) e inclua uma `Condition` que corresponda ao ID do aplicativo atribuído ao IdP. Exemplos de funções para diferentes provedores são descritos em Criar um perfil para um provedor de identidade de terceiros.

4. Em seu aplicativo, autentique seus usuários com o IdP. As especificações de como fazer isso variam de acordo com qual IdP você usa (Login with Amazon, Facebook ou Google) e em qual plataforma sua aplicação é executada. Por exemplo, o método de autenticação de um aplicativo Android pode ser diferente do de um aplicativo iOS ou de um aplicativo da web baseado em JavaScript.

   Normalmente, se o usuário ainda não estiver conectado, o IdP exibirá uma página de login. Depois que o IdP autentica o usuário, ele retorna um token de autenticação com informações sobre o usuário para o seu aplicativo. As informações incluídas dependem do que o IdP expõe e as informações que o usuário está disposto a compartilhar. Você pode usar essas informações em seu aplicativo.

5. Em seu aplicativo, faça uma chamada não assinada para a ação `AssumeRoleWithWebIdentity` para solicitar credenciais de segurança temporárias. Na solicitação, você passa o token de autenticação do IdP e especifica o nome do recurso da Amazon (ARN) para a função do IAM que você criou para esse IdP. A AWS verifica se o token é confiável e válido e, em caso afirmativo, retorna credenciais de segurança temporárias para sua aplicação com as permissões para a função que você indicou na solicitação. A resposta também inclui metadados sobre o usuário do IdP, como o ID de usuário exclusivo que o IdP associa ao usuário.

6. Usando as credenciais de segurança temporárias da resposta `AssumeRoleWithWebIdentity`, seu aplicativo faz solicitações assinadas às operações de API da AWS. As informações de ID do usuário do IdP podem diferenciar os usuários em sua aplicação. Por exemplo, você pode colocar objetos em pastas do Amazon S3 que incluam o ID do usuário como prefixos ou sufixos. Isso permite que você crie políticas de controle de acesso que bloqueiem a pasta para que somente o usuário com esse ID possa acessá-lo. Para obter mais informações, consulte AWS STSEntidades principais federadas do.

7. Seu aplicativo deve armazenar em cache as credenciais de segurança temporárias para que você não precise obter novas credenciais sempre que o aplicativo precisar fazer uma solicitação à AWS. Por padrão, as credenciais são válidas por uma hora. Quando as credenciais expiram (ou antes disso), você deve fazer outra chamada para `AssumeRoleWithWebIdentity` a fim de obter um novo conjunto de credenciais de segurança temporárias. Dependendo do IdP e como eles gerenciam seus tokens, talvez você precise atualizar o token do IdP antes de fazer uma nova chamada para `AssumeRoleWithWebIdentity`, pois os tokens do IdP geralmente também expiram após um período fixo. Se você usar o AWS SDK for iOS ou o AWS SDK for Android, poderá usar a ação AmazonSTSCredentialsProvider, que gerencia as credenciais temporárias do IAM, incluindo a atualização delas conforme necessário.

Federação OIDC

Imagine que você esteja criando uma aplicação que acesse recursos da AWS, como o GitHub Actions, que usa fluxos de trabalho para acessar o Amazon S3 e o DynamoDB.

Ao usar esses fluxos de trabalho, você fará solicitações para os serviços da AWS que devem ser assinadas com uma chave de acesso da AWS. No entanto, é altamente recomendável não armazenar credenciais de longo prazo da AWS em aplicações externas à AWS. Em vez disso, configure suas aplicações para solicitar credenciais de segurança temporárias da AWS dinamicamente quando necessário usando a federação OIDC. As credenciais temporárias fornecidas são mapeadas em um perfil da AWS que têm apenas as permissões necessárias para executar as tarefas solicitadas pela aplicação.

Com a federação OIDC, não é necessário criar códigos de início de sessão personalizados nem gerenciar suas próprias identidades de usuários. Em vez disso, você pode usar o OIDC em aplicações, como o GitHub Actions ou qualquer outro IdP compatível com o OpenID Connect (OIDC), para autenticar com o AWS. Elas recebem um token de autenticação, conhecido como JSON Web Token (JWT) e, em seguida, trocam esse token por credenciais de segurança temporárias na AWS que são mapeadas em um perfil do IAM com permissões para usar recursos específicos na sua Conta da AWS. O uso de um IdP ajuda você a manter sua Conta da AWS segura, pois não é necessário incorporar e distribuir credenciais de segurança de longo prazo com sua aplicação.

A federação OIDC oferece suporte à autenticação máquina a máquina (como pipelines de CI/CD, scripts automatizados e aplicações com tecnologia sem servidor) e à autenticação de usuário humano. Para cenários de autenticação de usuário humano em que você precisa gerenciar a inscrição, o login e os perfis de usuário, considere usar o Amazon Cognito como intermediador de identidades. Para obter detalhes sobre o uso do Amazon Cognito com o OIDC, consulte Amazon Cognito para aplicativos móveis.

> **Note**
>
> Os JSON Web Tokens (JWTs) emitidos pelos provedores de identidade do OpenID Connect (OIDC) contêm um prazo de validade na declaração `exp` que especifica quando o token expira. O IAM fornece uma janela de cinco minutos além do tempo de expiração especificado no JWT para considerar a distorção do relógio, conforme permitido pelo padrão OpenID Connect (OIDC) Core 1.0. Isso significa que os JWTs do OIDC recebidos pelo IAM após o prazo de expiração, mas dentro dessa janela de cinco minutos, são aceitos para avaliação e processamento adicionais.

Tópicos

- Recursos adicionais para federação OIDC
- Criar um provedor de identidade OpenID Connect (OIDC) no IAM
- Obter a impressão digital para um provedor de identidade OpenID Connect
- Controles de provedor de identidade para provedores OIDC compartilhados

Recursos adicionais para federação OIDC

Os seguintes recursos podem ajudar você a saber mais sobre a federação OIDC:

- Saiba como usar o OpenID Connect em seus fluxos de trabalho do GitHub em Configurar o OpenID Connect na Amazon Web Services
- Amazon Cognito Identity no Guia do Amplify Libraries para Android e Amazon Cognito Identity no Guia do Amplify Libraries para Swift.
- Como usar o ID externo ao conceder acesso aos seus recursos da AWS. O Blog de Segurança da AWS fornece orientação sobre como configurar com segurança o acesso entre contas e a federação de identidades externa.

Criar um provedor de identidade OpenID Connect (OIDC) no IAM

Provedores de identidade OIDC do IAM são entidades no IAM que descrevem um serviço de provedor de identidade (IdP) externo que oferece suporte ao padrão OpenID Connect (OIDC), como Google ou Salesforce. Você usa um provedor de identidade OIDC do IAM quando deseja estabelecer confiança entre um IdP compatível com OIDC e sua Conta da AWS. Isso será útil quando você estiver criando um aplicativo móvel ou um aplicativo web que precise acessar os recursos da AWS, mas não quiser criar um código de acesso personalizado ou gerenciar suas próprias identidades de usuários. Para ter mais informações sobre esse cenário, consulte the section called “Federação OIDC”.

Você pode criar e gerenciar um provedor de identidade OIDC do IAM usando o Console de gerenciamento da AWS, o AWS Command Line Interface, o Tools for Windows PowerShell ou a API do IAM.

Depois de criar um provedor de identidade OIDC do IAM, você deve criar uma ou mais funções do IAM. Função é uma identidade na AWS que não tem as próprias credenciais (como um usuário). Porém, neste contexto, um perfil é atribuído dinamicamente a uma entidade principal federada

que é autenticada pelo IdP da sua organização. A função permite que o IdP de sua organização solicite credenciais de segurança temporárias para acesso à AWS. As políticas atribuídas ao perfil determinam o que os usuários podem fazer na AWS. Para criar uma função para um provedor de identidade de terceiros, consulte Criar um perfil para um provedor de identidade de terceiros.

> **Important**
>
> Quando você configura políticas baseadas em identidade para ações que oferecem suporte a recursos do `oidc-provider`, o IAM avalia o URL completo do provedor de identidades OIDC, incluindo todos os caminhos especificados. Se o URL do seu provedor de identidades OIDC tiver um caminho, será necessário incluir esse caminho no ARN do `oidc-provider` como um valor de elemento `Resource`. Você também tem a opção de acrescentar uma barra e um curinga (`/*`) ao domínio do URL ou usar caracteres curinga (`*` e `?`) em qualquer ponto do caminho do URL. Se o URL do provedor de identidades OIDC na solicitação não corresponder ao valor definido no elemento `Resource` da política, a solicitação falhará.

Para solucionar problemas comuns com a federação OIDC do IAM, consulte Solucionar erros relacionados ao OIDC no re:Post AWS.

Tópicos

- Pré-requisitos: valide a configuração do seu provedor de identidade
- Criar e gerenciar um provedor OIDC (console)
- Criar e gerenciar um provedor de identidade OIDC do IAM (AWS CLI)
- Criar e gerenciar um provedor de identidade OIDC (API da AWS)

Pré-requisitos: valide a configuração do seu provedor de identidade

Antes de criar um provedor de identidade OIDC do IAM, é necessário ter as seguintes informações do IdP. Para obter mais informações sobre como obter informações de configuração do provedor OIDC, consulte a documentação do IdP.

1. Determine o URL publicamente disponível do seu provedor de identidade OIDC. O URL deve começar com https://. De acordo com o padrão OIDC, componentes de caminho são permitidos, mas parâmetros de consulta não são. Normalmente, o URL consiste apenas em um nome de host, como https://server.example.org ou https://example.com. O URL não deve conter um número de porta.

2. Adicione /.well-known/openid-configuration ao final do URL do provedor de identidade OIDC para ver o documento de configuração e os metadados disponíveis publicamente desse provedor. Você deve ter um documento de descoberta no formato JSON com o documento de configuração e os metadados do provedor que possam ser recuperados do URL do endpoint de descoberta do provedor OpenID Connect.
3. Confirme se os valores a seguir estão incluídos nas informações de configuração do provedor. Se algum desses campos estiver faltando em openid-configuration, você deverá atualizar seu documento de descoberta. Como esse processo pode variar dependendo do provedor de identidade, siga a documentação do seu IdP para concluir a tarefa.

   - issuer: o URL do seu domínio.
   - jwks_uri: o endpoint JSON Web Key Set (JWKS) em que o IAM obtém suas chaves públicas. Seu provedor de identidade deve incluir um endpoint JSON Web Key Set (JWKS) em openid-configuration. Esse URI define onde obter suas chaves públicas que são usadas para verificar os tokens assinados pelo provedor de identidade.

     > **Note**
     >
     > O JSON Web Key Set (JWKS) deve conter pelo menos uma chave e pode ter no máximo 100 chaves RSA e 100 chaves EC. Se o JWKS do provedor de identidades OIDC contiver mais de 100 chaves RSA ou 100 chaves EC, uma exceção `InvalidIdentityToken` será retornada quando a operação da API AssumeRoleWithWebIdentity for usada com um JWT assinado por um tipo de chave que exceda o limite de 100 chaves. Por exemplo, se um JWT for assinado com o algoritmo RSA e houver mais de 100 chaves RSA no JWKS do provedor, uma exceção `InvalidIdentityToken` será retornada.

   - claims_supported: informações sobre o usuário que ajudam a garantir que as respostas de autenticação OIDC do IdP contenham os atributos obrigatórios que a AWS utiliza em políticas do IAM para verificar as permissões de entidades principais federadas OIDC. Para obter uma lista das chaves de condição do IAM que podem ser usadas para declarações, consulte Chaves disponíveis para federação OIDC da AWS.
     - aud: você deve determinar o valor da declaração de audiência que seu IdP emite em JSON Web Tokens (JWTs). A declaração de audiência (aud) é específica da aplicação e identifica os destinatários pretendidos do token. Quando você registra uma aplicação Web ou móvel em um provedor OpenID Connect, este estabelece um ID de cliente que identifica a aplicação. O ID do cliente é um identificador exclusivo da sua aplicação, transmitido

na declaração aud para autenticação. A declaração aud deve corresponder ao valor de Audiência ao criar seu provedor de identidade OIDC do IAM.

- iat: as declarações devem incluir um valor para `iat` que represente a hora em que o token de ID é emitido.
- iss: o URL do provedor de identidade. O URL deve começar com https:// e corresponder ao URL do provedor fornecido ao IAM. De acordo com o padrão OIDC, componentes de caminho são permitidos, mas parâmetros de consulta não são. Normalmente, o URL consiste apenas em um nome de host, como https://server.example.org ou https://example.com. O URL não deve conter um número de porta.

- response_types_supported: id_token
- subject_types_supported: public
- id_token_signing_alg_values_supported: RS256, RS384, RS512, ES256, ES384, ES512

> **Note**
>
> Você pode incluir declarações adicionais, como `my_custom_claim` no exemplo abaixo. No entanto, o AWS STS ignorará a declaração.

```
{
  "issuer": "https://example-domain.com",
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "my_custom_claim"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256",
    "RS384",
    "RS512",
    "ES256",
    "ES384",
    "ES512"
```

```
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

Criar e gerenciar um provedor OIDC (console)

Siga estas instruções para criar e gerenciar um provedor de identidade OIDC do IAM no Console de gerenciamento da AWS.

> **Important**
>
> Se você estiver usando um provedor de identidade OIDC do Google, Facebook ou Amazon Cognito, não crie um provedor de identidade do IAM separado usando este procedimento. Esses provedores de identidade OIDC já estão integrados à AWS e estão disponíveis para uso. Em vez disso, siga as etapas para criar novas funções para seu provedor de identidade, consulte Criar uma função para uma federação do OpenID Connect (console).

Para criar um provedor de identidade OIDC do IAM (console)

1. Antes de criar um provedor de identidade OIDC do IAM, é necessário registrar sua aplicação com o IdP para receber um ID do cliente. O ID do cliente (também conhecido como público) é um identificador exclusivo para o seu aplicativo, emitido quando você registra o aplicativo com o IdP. Para obter mais informações sobre como obter um ID de cliente, consulte a documentação do IdP.

   > **Note**
   >
   > AWSA protege a comunicação com provedores de identidade (IdPs) OIDC usando nossa biblioteca de autoridades de certificação (CAs) raiz confiáveis para verificar o certificado TLS do endpoint JSON Web Key Set (JWKS). Se o seu IdP do OIDC depende de um certificado que não é assinado por uma dessas CAs confiáveis, somente então protegemos a comunicação usando as impressões digitais definidas na configuração do IdP. A AWS recorrerá à verificação de impressão digital se não conseguirmos recuperar o certificado TLS ou se o TLS v1.3 for necessário.

2. Abra o console do IAM em https://console.aws.amazon.com/iam/.
3. No painel de navegação, escolha Identity providers (Provedores de identidade) e, em seguida Add provider (Adicionar provedor).
4. Para Configure provider (Configurar provedor), escolha OpenID Connect.
5. Para URL do provedor, digite o URL do IdP. O URL deve estar em conformidade com estas restrições:

   - O URL diferencia maiúsculas de minúsculas.
   - O URL deve começar com **`https://`**.
   - O URL não deve conter um número de porta.
   - Dentro da sua Conta da AWS, cada provedor de identidade OIDC do IAM deve usar um URL exclusivo. Se você tentar enviar um URL que já foi usado por um provedor OpenID Connect no Conta da AWS, receberá um erro.

6. Em Audience (Público), digite o ID do cliente da aplicação que você registrou com o IdP e recebeu na Step 1 e que fará solicitações para a AWS. Se você tiver IDs do cliente adicionais (também conhecidos como públicos) para este IdP, poderá adicioná-los, posteriormente, na página de detalhes do provedor.

   > **Note**
   >
   > Se seu token JWT do IdP incluir a declaração `azp`, insira esse valor como o valor de Audiência.
   > Se seu provedor de identidade OIDC estiver definindo ambas as reivindicações no token `aud` e `azp`, o AWS STS usará o valor na declaração `azp` como a declaração `aud`.

7. (Opcional) Em Add tags (Adicionar etiquetas), você pode adicionar pares de chave-valor para ajudar na identificação e organização de seus IdPs. Você também pode usar tags para controlar o acesso aos recursos da AWS. Para saber mais sobre como etiquetar provedores de identidade OIDC do IAM, consulte Marcar provedores de identidades OpenID Connect (OIDC). Escolha Adicionar Tag. Insira valores para cada par de chave-valor de tag.
8. Verifique as informações fornecidas. Quando terminar, escolha Add provider (Adicionar provedor). O IAM tentará recuperar e usar a impressão digital da CA intermediária superior do certificado do servidor OIDC IdP para criar o provedor de identidade OIDC do IAM.

> **Note**
>
> A cadeia de certificados do provedor de identidade de OIDC deverá começar com o URL do domínio ou do emissor, depois com o certificado intermediário e terminar com o certificado raiz. Se a ordem da cadeia de certificados for diferente ou incluir certificados duplicados ou adicionais, você receberá um erro de incompatibilidade de assinatura e o STS falhará ao validar o JSON Web Token (JWT). Corrija a ordem dos certificados na cadeia retornados do servidor para resolver o erro. Para obter mais informações sobre os padrões da cadeia de certificados, consulte certificate_list in RFC 5246 no site RFC Series.

9. Atribua uma função do IAM ao seu provedor de identidade para fornecer identidades de usuário externo gerenciadas pelo seu provedor de identidade, permissões para acessar recursos da AWS em sua conta. Para saber mais sobre como criar funções para a federação de identidades, consulte Criar um perfil para um provedor de identidade de terceiros.

> **Note**
>
> Os IdPs OIDC usados em uma política de confiança de perfil devem estar na mesma conta que o perfil que confia nela.

Para adicionar ou remover uma impressão digital de um provedor de identidade OIDC do IAM (console)

> **Note**
>
> AWSA protege a comunicação com provedores de identidade (IdPs) OIDC usando nossa biblioteca de autoridades de certificação (CAs) raiz confiáveis para verificar o certificado TLS do endpoint JSON Web Key Set (JWKS). Se o seu IdP do OIDC depende de um certificado que não é assinado por uma dessas CAs confiáveis, somente então protegemos a comunicação usando as impressões digitais definidas na configuração do IdP. A AWS recorrerá à verificação de impressão digital se não conseguirmos recuperar o certificado TLS ou se o TLS v1.3 for necessário.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação, escolha Identity providers (Provedores de identidade). Em seguida, escolha o nome do provedor de identidade do IAM que você deseja atualizar.
3. Escolha a guia Verificação de endpoint e, na seção Impressões digitais, escolha Gerenciar. Para inserir um novo valor de impressão digital, escolha Add thumbprint (Adicionar impressão digital). Para remover uma impressão digital, escolha Remove (Remover) ao lado do item que deseja remover.

   > **Note**
   >
   > Um provedor de identidade OIDC do IAM deve ter, pelo menos, uma impressão digital e, no máximo, cinco impressões digitais.

   Quando concluir, escolha Save changes (Salvar alterações).

Para adicionar um público para um provedor de identidade OIDC do IAM (console)

1. No painel de navegação, escolha Identity providers (Provedores de identidade) e escolha o nome do provedor de identidade do IAM que você deseja atualizar.
2. Na seção Audiences (Público-alvo), escolha Actions (Ações) e selecione Add audience (Adicionar público-alvo).
3. Digite o ID do cliente da aplicação que você registrou com o IdP e recebeu em Step 1 e que fará solicitações para a AWS. Em seguida, escolha Add audiences (Adicionar público-alvo).

   > **Note**
   >
   > Um provedor de identidade OIDC do IAM deve ter pelo menos um público e, no máximo, 100 públicos.

Para remover um público de um provedor de identidade OIDC do IAM (console)

1. No painel de navegação, escolha Identity providers (Provedores de identidade) e escolha o nome do provedor de identidade do IAM que você deseja atualizar.
2. Na seção Audiences (Público-alvo), selecione o botão de opção ao lado do público-alvo que você deseja remover e, em seguida, selecione Actions (Ações).

3. Escolha Remove audience (Remover público-alvo). Uma nova janela é aberta.
4. Se você remover um público-alvo, as identidades federadas com o público não poderão assumir funções associadas ao público. Na janela, leia o aviso e confirme se deseja remover o público-alvo digitando a palavra `remove` no campo.
5. Escolha Remove (Remover) para remover o público-alvo.

Para excluir um provedor de identidade OIDC do IAM (console)

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Identity providers (Provedores de identidade).
3. Marque a caixa de seleção ao lado do provedor de identidade do IAM que você deseja excluir. Uma nova janela é aberta.
4. Confirme se deseja excluir o provedor digitando a palavra `delete` no campo. Em seguida, selecione Excluir.

Criar e gerenciar um provedor de identidade OIDC do IAM (AWS CLI)

Você pode usar os comandos da AWS CLI a seguir para criar e gerenciar provedores de identidade OIDC do IAM.

Para criar um provedor de identidade OIDC do IAM (AWS CLI)

1. (Opcional) Para obter uma lista de todos os provedores de identidade OIDC do IAM na sua conta da AWS, execute o seguinte comando:
   - `aws iam list-open-id-connect-providers`
2. Para criar um novo provedor de identidade OIDC do IAM, execute o seguinte comando:
   - `aws iam create-open-id-connect-provider`

Para atualizar a lista de impressões digitais de certificado do servidor de um provedor de identidade OIDC do IAM existente (AWS CLI)

- Para atualizar a lista de impressões digitais de certificado do servidor de um provedor de identidade OIDC do IAM, execute o seguinte comando:
  - `aws iam update-open-id-connect-provider-thumbprint`

Para etiquetar um provedor de identidade OIDC do IAM existente (AWS CLI)

- Para etiquetar um provedor de identidade OIDC do IAM existente, execute o seguinte comando:
  - `aws iam tag-open-id-connect-provider`

Para listar etiquetas para um provedor de identidade OIDC do IAM existente (AWS CLI)

- Para listar etiquetas para um provedor de identidade OIDC do IAM existente, execute o seguinte comando:
  - `aws iam list-open-id-connect-provider-tags`

Para remover etiquetas em um provedor de identidade OIDC do IAM (AWS CLI)

- Para remover etiquetas de um provedor de identidade OIDC do IAM existente, execute o seguinte comando:
  - `aws iam untag-open-id-connect-provider`

Para adicionar ou remover um ID de cliente de um provedor de identidade OIDC do IAM existente (AWS CLI)

1. (Opcional) Para obter uma lista de todos os provedores de identidade OIDC do IAM na sua conta da AWS, execute o seguinte comando:
   - `aws iam list-open-id-connect-providers`
2. (Opcional) Para obter informações detalhadas sobre um provedor de identidade OIDC do IAM, execute o seguinte comando:
   - `aws iam get-open-id-connect-provider`
3. Para adicionar um novo ID do cliente a um provedor de identidade OIDC do IAM existente, execute o seguinte comando:
   - `aws iam add-client-id-to-open-id-connect-provider`
4. Para remover um cliente de um provedor de identidade OIDC do IAM existente, execute o seguinte comando:

- `aws iam remove-client-id-from-open-id-connect-provider`

Para excluir um provedor de identidade OIDC do IAM (AWS CLI)

1. (Opcional) Para obter uma lista de todos os provedores de identidade OIDC do IAM na sua conta da AWS, execute o seguinte comando:
   - `aws iam list-open-id-connect-providers`
2. (Opcional) Para obter informações detalhadas sobre um provedor de identidade OIDC do IAM, execute o seguinte comando:
   - `aws iam get-open-id-connect-provider`
3. Para excluir um provedor de identidade OIDC do IAM, execute o seguinte comando:
   - `aws iam delete-open-id-connect-provider`

Criar e gerenciar um provedor de identidade OIDC (API da AWS)

Você pode usar os comandos da API do IAM a seguir para criar e gerenciar provedores OIDC.

Para criar um provedor de identidade OIDC do IAM (API da AWS)

1. (Opcional) Para obter uma lista de todos os provedores de identidade OIDC do IAM em sua conta da AWS, chame a seguinte operação:
   - `ListOpenIDConnectProviders`
2. Para criar um novo provedor de identidade OIDC do IAM, chame a seguinte operação:
   - `CreateOpenIDConnectProvider`

Para atualizar a lista de impressões digitais de certificado do servidor de um provedor de identidade OIDC do IAM existente (API da AWS)

- Para atualizar a lista de impressões digitais de certificado do servidor de um provedor de identidade OIDC do IAM, chame a seguinte operação:
  - `UpdateOpenIDConnectProviderThumbprint`

Para etiquetar um provedor de identidade OIDC do IAM existente (API da AWS)

- Para etiquetar um provedor de identidade OIDC do IAM existente, chame a seguinte operação:
  - `TagOpenIDConnectProvider`

Para listar etiquetas para um provedor de identidade OIDC do IAM existente (API da AWS)

- Para listar etiquetas para um provedor de identidade OIDC do IAM existente, chame a seguinte operação:
  - `ListOpenIDConnectProviderTags`

Para remover etiquetas em um provedor de identidade OIDC do IAM existente (API da AWS)

- Para remover etiquetas de um provedor de identidade OIDC do IAM existente, chame a seguinte operação:
  - `UntagOpenIDConnectProvider`

Para adicionar ou remover um ID de cliente de um provedor de identidade OIDC do IAM existente (API da AWS)

1. (Opcional) Para obter uma lista de todos os provedores de identidade OIDC do IAM em sua conta da AWS, chame a seguinte operação:
   - `ListOpenIDConnectProviders`
2. (Opcional) Para obter informações detalhadas sobre um provedor de identidade OIDC do IAM, chame a seguinte operação:
   - `GetOpenIDConnectProvider`
3. Para adicionar um novo ID do cliente a um provedor de identidade OIDC do IAM existente, chame a seguinte operação:
   - `AddClientIDToOpenIDConnectProvider`
4. Para remover um ID do cliente de um provedor de identidade OIDC do IAM existente, chame a seguinte operação:

- `RemoveClientIDFromOpenIDConnectProvider`

Para excluir um provedor de identidade OIDC do IAM (API da AWS)

1. (Opcional) Para obter uma lista de todos os provedores de identidade OIDC do IAM em sua conta da AWS, chame a seguinte operação:
   - `ListOpenIDConnectProviders`
2. (Opcional) Para obter informações detalhadas sobre um provedor de identidade OIDC do IAM, chame a seguinte operação:
   - `GetOpenIDConnectProvider`
3. Para excluir um provedor de identidade OIDC do IAM, chame a seguinte operação:
   - `DeleteOpenIDConnectProvider`

Obter a impressão digital para um provedor de identidade OpenID Connect

Quando você cria um provedor de identidade OpenID Connect (OIDC) no IAM, o IAM exige uma impressão digital para a autoridade de certificação (CA) intermediária superior que assinou o certificado usado pelo provedor de identidades (IdP). A impressão digital é uma assinatura para o certificado da CA que foi usada para emitir o certificado para o IdP compatível com OIDC. Ao criar um provedor de identidade OIDC do IAM, você confia nas identidades autenticadas por esse IdP para ter acesso à sua Conta da AWS. Ao usar a impressão digital do certificado da CA, você confiará em qualquer certificado emitido pela CA que tenha o mesmo nome do DNS registrado. Isso elimina a necessidade de atualizar confianças em cada conta quando você renovar o certificado de assinatura do IdP.

> **Important**
>
> Na maioria dos casos, o servidor de federação usa dois certificados diferentes:
>
> - O primeiro estabelece uma conexão HTTPS entre a AWS e seu IdP. Isso deve ser emitido por uma CA de raiz pública confiável, como o Gerenciador de certificados da AWS. Isso permite que o cliente verifique a confiabilidade e o status do certificado.

- O segundo é usado para criptografar tokens e deve ser assinado por uma CA de raiz privada ou pública.

É possível criar um provedor de identidade OIDC do IAM com a AWS Command Line Interface, o Tools for Windows PowerShell ou a API do IAM. Ao usar esses métodos, você tem a opção de fornecer manualmente uma impressão digital. Se você optar por não incluir uma impressão digital, o IAM recuperará a impressão digital da CA intermediária superior do certificado do servidor do IdP OIDC. Se você optar por incluir uma impressão digital, será necessário obtê-la manualmente e fornecê-la para a AWS.

Quando você cria um provedor de identidade OIDC com o console do IAM, o IAM tenta recuperar a impressão digital da CA intermediária superior do certificado de servidor IdP do OIDC para você.

Recomendamos que você também obtenha a impressão digital para o seu IdP OIDC manualmente e verifique se a impressão digital correta foi recuperada pelo IAM. Para obter mais informações sobre como obter impressões digitais do certificado, consulte as seções a seguir.

Note

AWSA protege a comunicação com provedores de identidade (IdPs) OIDC usando nossa biblioteca de autoridades de certificação (CAs) raiz confiáveis para verificar o certificado TLS do endpoint JSON Web Key Set (JWKS). Se o seu IdP do OIDC depende de um certificado que não é assinado por uma dessas CAs confiáveis, somente então protegemos a comunicação usando as impressões digitais definidas na configuração do IdP. A AWS recorrerá à verificação de impressão digital se não conseguirmos recuperar o certificado TLS ou se o TLS v1.3 for necessário.

Important

Se seu provedor de identidade OIDC usa hosts diferentes para o endpoint de descoberta (o URL do emissor) e o endpoint JWKS (`jwks_uri`), você deve incluir a impressão digital de ambos os endpoints na lista de impressões digitais do seu provedor OIDC. A AWS valida a impressão digital de ambos os terminais ao retornar à verificação da impressão digital. Se alguma das impressões digitais estiver ausente na lista configurada, a autenticação falhará.

A maioria dos provedores de OIDC hospeda os dois endpoints no mesmo domínio, portanto, uma única impressão digital inclui os dois. Você só precisa adicionar uma impressão digital separada se o provedor usar hosts ou certificados diferentes para os dois endpoints.

Obter impressão digital do certificado

É possível usar um navegador da Web e a ferramenta de linha de comando OpenSSL para obter a impressão digital do certificado para um provedor OIDC. No entanto, não é necessário obter manualmente a impressão digital do certificado para criar um provedor de identidade OIDC do IAM. É possível usar o seguinte procedimento para obter a impressão digital do certificado do provedor OIDC.

Para obter a impressão digital para um IdP OIDC

1. A fim de obter a impressão digital para um IdP OIDC, você precisa obter a ferramenta de linha de comando OpenSSL. Use esta ferramenta para baixar a cadeia de certificados do IdP OIDC e gerar uma impressão digital do certificado final na cadeia de certificados. Se você precisar instalar e configurar o OpenSSL, siga as instruções em Instalar o OpenSSL e Configurar o OpenSSL.
2. Comece com o URL do IdP OIDC (por exemplo, `https://server.example.com`) e, em seguida, adicione `/.well-known/openid-configuration` para formar o URL para o documento de configuração do IdP, como o seguinte:

   `https://`*`server.example.com`*`/.well-known/openid-configuration`

   Abra este URL em um navegador da Web, substituindo *`server.example.com`* pelo nome do seu servidor IdP.
3. No documento exibido, use o recurso Find (Localizar) do navegador da Web para localizar o texto `"jwks_uri"`. Imediatamente após o texto `"jwks_uri"` você verá dois pontos (:) seguidos por um URL. Copie o nome de domínio totalmente qualificado do URL. Não inclua `https://` nem qualquer caminho que venha após o domínio de nível superior.

   ```
   {
    "issuer": "https://accounts.example.com",
    "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
    "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
    "token_endpoint": "https://oauth2.exampleapis.com/token",
    "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
   ```

```
 "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
 "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
...
```

> **Note**
>
> Se seus endpoints de JWKS e descoberta usarem hosts diferentes, anote o nome de domínio totalmente qualificado do URL do emissor (o URL em que você usou em Step 2, sem o caminho `/.well-known/openid-configuration`). Se esse domínio for diferente do domínio `jwks_uri` que você copiou nesta etapa, execute Step 4 até Step 6 para os dois domínios e inclua as duas impressões digitais ao criar seu provedor OIDC.

4. Use a ferramenta de linha de comando OpenSSL para executar o comando a seguir. Substitua *keys.example.com* pelo nome de domínio que você obteve em Step 3.

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. Na janela de comando, role até ver um certificado semelhante ao exemplo a seguir. Se você vir mais de um certificado, encontre o último certificado exibido (no final da saída do comando). Ele contém o certificado da principal CA intermediária na cadeia de autoridade de certificação.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
 NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
 -----END CERTIFICATE-----
```

Copie o certificado (incluindo as linhas `-----BEGIN CERTIFICATE-----` e `-----END CERTIFICATE-----`) e cole-o em um arquivo de texto. Em seguida, salve o arquivo com o nome **`certificate.crt`**.

> **Note**
>
> A cadeia de certificados do provedor de identidade de OIDC deverá começar com o URL do domínio ou do emissor, incluir quaisquer certificados intermediários (caso presentes) e terminar com o certificado raiz. Se a ordem da cadeia de certificados for diferente ou incluir certificados duplicados ou adicionais, você receberá um erro de incompatibilidade de assinatura e o STS falhará ao validar o JSON Web Token (JWT). Corrija a ordem dos certificados na cadeia retornados do servidor para resolver o erro. Para obter mais informações sobre os padrões da cadeia de certificados, consulte certificate_list in RFC 5246 no site RFC Series.

6. Use a ferramenta de linha de comando OpenSSL para executar o comando a seguir.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

Sua janela de comando exibe a miniatura do certificado, que é similar ao exemplo a seguir:

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Remova os caracteres de dois pontos (:) desta string para produzir a impressão digital final, desta forma:

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

> **Note**
>
> Se seu endpoint de descoberta usa um host diferente do seu endpoint JWKS, repita Step 4 até Step 6 usando o domínio do endpoint de descoberta (seu domínio de URL do emissor). Inclua as duas impressões digitais ao criar ou atualizar seu provedor de identidade OIDC usando CreateOpenIDConnectProvider ou UpdateOpenIDConnectProviderThumbprint.

7. Se você estiver criando o provedor de identidade OIDC do IAM com a AWS CLI, o Tools for Windows PowerShell ou a API do IAM, é opcional fornecer uma impressão digital. Se você optar por não incluir uma impressão digital durante a criação, o IAM recuperará a impressão digital da CA intermediária superior do certificado do servidor do IdP OIDC. Depois que o provedor de identidade OIDC do IAM for criado, você poderá comparar essa impressão digital com a impressão digital recuperada pelo IAM.

   Se você estiver criando o provedor de identidade OIDC do IAM no console do IAM, o console tenta recuperar a impressão digital da CA intermediária superior do certificado de servidor IdP do OIDC para você. É possível comparar essa impressão digital com a impressão digital recuperada pelo IAM. Depois que o provedor de identidade OIDC do IAM for criado, você poderá ver a impressão digital do provedor de identidade OIDC do IAM na guia Verificação de endpoint na página do console Resumo do provedor OIDC.

   > **Important**
   >
   > Se a impressão digital obtida não corresponder à que você vir nos detalhes de impressão digital do provedor de identidade OIDC do IAM, você não deve usar o provedor OIDC. Em vez disso, você deve excluir o provedor OIDC criado e tentar criar o provedor OIDC novamente depois de um tempo. Verifique se as impressões digitais coincidem antes de usar o provedor. Se as impressões digitais ainda não coincidirem após uma segunda tentativa, use o Fórum do IAM para entrar em contato com a AWS.

Instalar o OpenSSL

Se você ainda não tiver o OpenSSL instalado, siga as instruções descritas nesta seção.

Para instalar o OpenSSL no Linux ou Unix

1. Acesse OpenSSL: Source, Tarballs (https://openssl.org/source/).
2. Baixe a fonte mais recente e crie o pacote.

Para instalar o OpenSSL no Windows

1. AcesseOpenSSL: distribuições binárias(https://wiki.openssl.org/index.php/Binaries) para obter uma lista de sites por meio dos quais você pode instalar a versão do Windows.
2. Siga as instruções no site selecionado para iniciar a instalação.

3. Se for solicitado que você instale os Redistribuíveis do Microsoft Visual C++ 2008 e eles ainda não estiverem instalados no seu sistema, escolha o link de download apropriado para o seu ambiente. Siga as instruções fornecidas pelo Assistente de instalação de redistribuível do Microsoft Visual C++ 2008.

   > **Note**
   >
   > Caso não tenha certeza de que os redistribuíveis do Microsoft Visual C++ 2008 já estão instalados no seu sistema, você poderá tentar instalar o OpenSSL primeiro. O instalador do OpenSSL exibe um alerta se os redistribuíveis do Microsoft Visual C++ 2008 ainda não estiverem instalados. Instale a arquitetura (32 bits ou 64 bits) que corresponde à versão do OpenSSL que você instalou.

4. Depois de ter instalado os redistribuíveis do Microsoft Visual C++ 2008, selecione a versão apropriada dos binários OpenSSL para o seu ambiente e salve o arquivo localmente. Inicie o Assistente de configuração do OpenSSL.
5. Siga as instruções descritas no OpenSSL Setup Wizard (Assistente de configuração do OpenSSL).

Configurar o OpenSSL

Antes de usar comandos OpenSSL, você deve configurar o sistema operacional para que ele tenha informações sobre o local em que o OpenSSL está instalado.

Para configurar o OpenSSL no Linux ou no Unix

1. Na linha de comando, defina a variável `OpenSSL_HOME` para o local da instalação do OpenSSL:

   ```
   $ export OpenSSL_HOME=path_to_your_OpenSSL_installation
   ```

2. Defina o caminho para incluir a instalação do OpenSSL:

   ```
   $ export PATH=$PATH:$OpenSSL_HOME/bin
   ```

   > **Note**
   >
   > Todas as alterações feitas nas variáveis de ambiente com o comando `export` são válidas apenas para a sessão atual. É possível fazer alterações persistentes nas

variáveis de ambiente definindo-as no arquivo de configuração do shell. Para obter mais informações, consulte a documentação do seu sistema operacional.

Para configurar o OpenSSL no Windows

1. Abra a janela Command Prompt (Prompt de comando).
2. Defina a variável `OpenSSL_HOME` para o local da instalação do OpenSSL:

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Defina a variável `OpenSSL_CONF` para o local do arquivo de configuração em sua instalação do OpenSSL:

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Defina o caminho para incluir a instalação do OpenSSL:

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Todas as alterações feitas nas variáveis de ambiente do Windows em uma janela Command Prompt (Prompt de comando) são válidas apenas para a sessão de linha de comando atual. É possível fazer alterações persistentes nas variáveis de ambiente definindo-as como propriedades do sistema. Os procedimentos exatos dependem da versão do Windows que você está usando. (Por exemplo, no Windows 7, abra Control Panel (Painel de Controle), System and Security (Sistema e Segurança), System (Sistema). Em seguida, escolha Advanced system settings (Configurações avançadas do sistema), guia Advanced (Avançado), Environment Variables (Variáveis de ambiente).) Para obter mais informações, consulte a documentação do Windows.

Controles de provedor de identidade para provedores OIDC compartilhados

Para provedores de identidade (IdPs) OpenID Connect (OIDC) compartilhados reconhecidos, o IAM exige uma avaliação explícita de declarações específicas nas políticas de confiança de perfis. Essas declarações obrigatórias, chamadas de controles de provedor de identidade, são avaliadas pelo

IAM durante a criação do perfil e as atualizações da política de confiança. Se a política de confiança do perfil não avaliar os controles exigidos pelo IdP OIDC compartilhado, a criação ou atualização do perfil falhará. Isso garante que somente identidades autorizadas da organização pretendida possam assumir perfis e acessar recursos da AWS. Esse controle de segurança é crucial quando os provedores OIDC são compartilhados entre vários clientes da AWS.

Os controles do provedor de identidade não serão avaliados pelo IAM para as políticas de confiança de perfis OIDC existentes. Para quaisquer modificações na política de confiança do perfil para perfis OIDC existentes, o IAM exigirá que os controles do provedor de identidade sejam incluídos na política de confiança do perfil.

Tipos de provedores OIDC

O IAM categoriza os provedores de identidade OIDC em dois tipos distintos: privados e compartilhados. Um IdP OIDC privado pode pertencer e ser gerenciado por uma única organização ou pode ser um locatário de um provedor de SaaS, com seu URL do emissor OIDC servindo como um identificador exclusivo específico para essa organização. Por outro lado, um IdP OIDC compartilhado é utilizado por várias organizações, onde o URL do emissor OIDC pode ser idêntico para todas as organizações que usam esse provedor de identidade compartilhado.

A tabela abaixo descreve as principais diferenças entre provedores OIDC privados e compartilhados:

Característica	Provedor OIDC privado	Provedor OIDC compartilhado
Emissor	Exclusivo para a organização	Compartilhado entre várias organizações
Informações de locação	Comunicado por meio de emissor exclusivo	Comunicado por meio de declarações no JWT
Requisitos da política de confiança	Não há necessidade de avaliação da declaração específica	A avaliação de declarações específicas é necessária

Provedores de identidade OIDC compartilhados com controles de provedores de identidade

Quando você cria ou modifica um provedor OIDC no IAM, o sistema identifica e avalia automaticamente as declarações necessárias para provedores OIDC compartilhados reconhecidos.

Se os controles do provedor de identidade não estiverem configurados na política de confiança do perfil, a criação ou atualização do perfil falhará com um erro MalformedPolicyDocument.

A tabela a seguir lista os provedores OIDC compartilhados que exigem controles de provedores de identidades nas políticas de confiança de perfis e informações adicionais para ajudar a configurar provedores de identidades.

IdP OIDC	URL OIDC	Declaração de locação	Declarações necessárias
Amazon Cognito	`cognito-identity.amazonaws.com`	aud	`cognito-identity.amazonaws.com:aud`
Azure Sentinel	`https://sts.windows.net/33e01921-4d64-4f8c-a055-5bdaffd5e33d`	`sts:RoleSessionName`	`sts:RoleSessionName`
Buildkite	`https://agent.buildkite.com`	sub	`agent.buildkite.com:sub`
Codefresh SaaS	`https://oidc.codefresh.io`	sub	`oidc.codefresh.io:sub`
DVC Studio	`https://studio.datachain.ai/api`	sub	`studio.datachain.ai/api:sub`
Ações do GitHub	`https://token.actions.githu`	sub ou job_workflow_ref	`token.actions.gith`

IdP OIDC	URL OIDC	Declaração de locação	Declarações necessárias

	`buserconte nt.com`		`ubusercon tent.com:sub`
			
			ou
			
			`token.act ions.gith ubusercon tent.com: job_workf low_ref`

Streaming do log de auditoria do GitHub	`https://oidc- configuration. audit-log .githubus ercontent.com`	sub	`oidc-conf iguration .audit-lo g.githubu serconten t.com:sub`
vstoken GitHub	`https://v stoken.ac tions.git hubuserco ntent.com`	sub	`vstoken.a ctions.gi thubuserc ontent.com:sub`
GitLab	`https://g itlab.com`	sub	`gitlab.com:sub`

IdP OIDC	URL OIDC	Declaração de locação	Declarações necessárias
IBM Turbonomic SaaS*	• https://rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08iidd0nt4hekbn7 • https://rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr • https://rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg • https://rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26	sub	• rh-oidc.s3.us-east-1.amazonaws.com/22ejnvnnturfmt6km08idd0nt4hekbn7:sub • rh-oidc.s3.us-east-1.amazonaws.com/23e3sd27sju1hoou6ohfs68vbno607tr:sub • rh-oidc.s3.us-east-1.amazonaws.com/23ne21h005qjl3n33d8dui5dlrmv2tmg:sub • rh-oidc.s3.us-east-1.amazonaws.com/24jrf12m5dj7ljlfb4ta2frhrcoadm26:sub

IdP OIDC	URL OIDC	Declaração de locação	Declarações necessárias
	• https://oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb • https://oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu		• oidc.op1.openshiftapps.com/2f785sojlpb85i7402pk3qogugim5nfb:sub • oidc.op1.openshiftapps.com/2c51blsaqa9gkjt0o9rt11mle8mmropu:sub
Pulumi Cloud	https://api.pulumi.com/oidc	aud	api.pulumi.com/oidc:aud
sandboxes.cloud	https://sandboxes.cloud	aud	sandboxes.cloud:aud
Scalr	https://scalr.io	sub	scalr.io:sub
Shisho Cloud	https://tokens.cloud.shisho.dev	sub	tokens.cloud.shisho.dev:sub
Terraform Cloud	https://app.terraform.io	sub	app.terraform.io:sub

IdP OIDC	URL OIDC	Declaração de locação	Declarações necessárias
Upbound	`https://proidc.upbound.io`	sub	`proidc.upbound.io:sub`
Endpoint global do Vercel	`https://oidc.vercel.com`	aud	`oidc.vercel.com:aud`

* O IBM Turbonomic atualiza periodicamente seu URL de emissor do OIDC com novas versões da plataforma. Adicionaremos outros emissores OIDC do Turbonomic no escopo como um provedor compartilhado conforme necessário.

Para quaisquer novos IdPs OIDC que o IAM identifique como compartilhados, os controles de provedor de identidade necessários para as políticas de confiança de perfis serão documentados e aplicados de maneira semelhante.

Recursos adicionais

Recursos adicionais:

- Para obter mais informações sobre como criar um perfil IAM para federação OIDC, consulte Criar uma função para uma federação do OpenID Connect (console).
- Para obter uma lista das chaves de condição do IAM que podem ser usadas para declarações, consulte Chaves disponíveis para federação OIDC da AWS.

Federação SAML 2.0

A AWS dá suporte à federação de identidades com o SAML 2.0 (Security Assertion Markup Language 2.0), um padrão aberto que muitos provedores de identidade (IdPs) usam. Esse recurso permite a autenticação única (SSO) federada, para que os usuários possam fazer login no Console de gerenciamento da AWS ou chamar as operações de API da AWS sem que você precise criar um usuário do IAM para todos em sua organização. Ao usar o SAML, simplifique o processo de configuração da federação com a AWS, pois poderá usar o serviço do IdP, em vez de gravar o código de proxy de identidade personalizado.

> **Note**
>
> A federação de identidade SAML do IAM é compatível com respostas SAML criptografadas de provedores de identidade (IdPs) federados baseados em SAML. O IAM Identity Center e o Amazon Cognito não são compatíveis com declarações de SAML criptografadas provenientes de provedores de identidade SAML do IAM.
> É possível adicionar indiretamente compatibilidade com declarações de SAML criptografadas à federação do banco de identidades do Amazon Cognito com grupos de usuários do Amazon Cognito. Os grupos de usuários têm uma federação SAML que é independente da federação SAML do IAM e é compatível com assinatura e criptografia de SAML. Embora esse atributo não se estenda diretamente aos bancos de identidades, os grupos de usuários podem ser IdPs dos bancos de identidades. Para usar a criptografia SAML com bancos de identidades, adicione um provedor de SAML com criptografia a um grupo de usuários que seja um IdP de um banco de identidades.
> Seu provedor de SAML deverá ser capaz de criptografar declarações de SAML com uma chave fornecida pelo seu grupo de usuários. Os grupos de usuários não aceitarão declarações criptografadas com um certificado fornecido pelo IAM.

A federação do IAM é compatível com estes casos de uso:

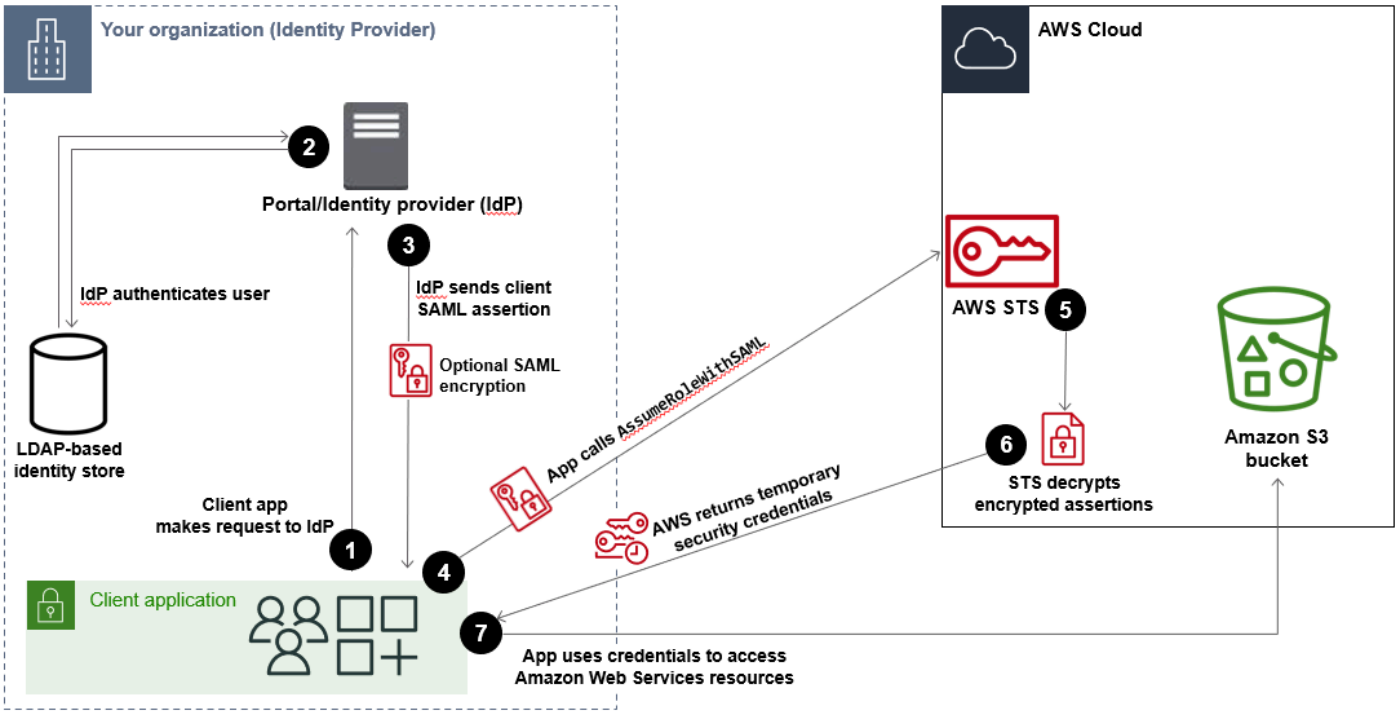
- Acesso federado para permitir que um usuário ou aplicativo na sua organização chame operações de API da AWS. Esse caso de uso é abordado na seção a seguir. Use uma declaração do SAML (como parte da resposta de autenticação) gerada na sua organização para obter credenciais de segurança temporárias. Este cenário é semelhante a outros cenários de federação compatíveis com o IAM, como os descritos em Solicitar credenciais de segurança temporárias e Federação OIDC. No entanto, os IdPs baseados no SAML 2.0 da sua organização lidam com muitos detalhes em tempo de execução para realizar a verificação de autenticação e autorização.
- Logon único (SSO) baseado na web para o Console de gerenciamento da AWS da sua organização. Os usuários podem entrar em um portal em sua organização hospedado por um IdP compatível com SAML 2.0, selecionar uma opção para acessar a AWS e ser redirecionado para o console sem precisar fornecer informações de login adicionais. Você pode usar um IdP do SAML de terceiros para estabelecer acesso ao SSO para o console ou pode criar um IdP personalizado para habilitar o acesso ao console para os usuários externos. Para obter mais informações sobre a criação de um IdP personalizado, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.

Tópicos

- Usar a federação baseada em SAML para acesso da API à AWS
- Visão geral da configuração de federação baseada em SAML 2.0
- Visão geral da função para permitir acesso federado do SAML aos seus recursos da AWS
- Identificar exclusivamente os usuários na federação baseada em SAML
- Criar um provedor de identidades SAML no IAM
- Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações
- Integrar provedores de soluções SAML de terceiros com a AWS
- Configurar declarações SAML para a resposta de autenticação
- Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS
- Visualizar uma resposta do SAML no navegador

Usar a federação baseada em SAML para acesso da API à AWS

Imagine que você deseja fornecer uma maneira para os funcionários copiarem dados de seus computadores para uma pasta de backup. Crie um aplicativo que possa ser executado pelos usuários em seus computadores. No backend, a aplicação lê e grava objetos em um bucket do Amazon S3. Os usuários não têm acesso direto à AWS. Em vez disso, o processo seguinte é usado:



1. Um usuário na sua organização usa um aplicativo do cliente para solicitar a autenticação do IdP da sua organização.
2. O IdP autentica o usuário em relação ao armazenamento de identidades da sua organização.
3. O IdP cria uma declaração do SAML com informações sobre o usuário e a envia para o aplicativo do cliente. Quando você habilita a criptografia de SAML para um IdP SAML do IAM, essa declaração é criptografada por seu IdP externo.
4. O aplicativo do cliente chama a API do AWS STS `AssumeRoleWithSAML`, transmitindo o ARN do provedor SAML, o ARN da função a ser assumida e a declaração do SAML do IdP. Se a criptografia estiver habilitada, a declaração transmitida por meio da aplicação cliente permanecerá criptografada em trânsito.
5. (Opcional) O AWS STS usa a chave privada que você carregou do seu IdP externo para descriptografar a declaração SAML criptografada.
6. A resposta da API para o aplicativo do cliente inclui credenciais de segurança temporárias.
7. A aplicação cliente usa as credenciais de segurança temporárias para chamar as operações de API do Amazon S3.

Visão geral da configuração de federação baseada em SAML 2.0

Antes de usar a federação baseada em SAML 2.0, como descrito no diagrama e no cenário anterior, configure seu IdP da organização e sua Conta da AWS para confiarem uns nos outros. O processo geral para configurar essa confiança é descrito nas etapas a seguir. Dentro da sua organização, é necessário ter um IdP que seja compatível com SAML 2.0, como o Microsoft Active Directory Federation Service (AD FS, parte do Windows Server), Shibboleth ou outro provedor compatível com o SAML 2.0.

> **Note**
>
> Para melhorar a resiliência da federação, recomendamos que você configure seu IdP e sua federação da AWS para oferecer suporte a vários endpoints de login do SAML. Para obter detalhes, consulte o artigo do AWS Security Blog How to use regional SAML endpoints for failover.

Configurar o IdP da sua organização e a AWS para confiança mútua

1. Registre a AWS como provedora de serviços (SP) com o IdP da sua organização. Use o documento de metadados SAML de `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

   Para obter uma lista dos possíveis valores de *region-code* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS).

   Ou você também pode usar o documento de metadados SAML em `https://signin.aws.amazon.com/static/saml-metadata.xml`.

2. Ao usar o IdP da sua organização, você gera um arquivo XML de metadados do SAML equivalente que pode descrever o IdP como um provedor de identidade do IAM na AWS. Ele deve incluir o nome do emissor, uma data de criação, uma data de expiração e chaves que a AWS pode usar para validar as respostas de autenticação (declarações) da sua organização.

   Se você permitir o envio de declarações SAML criptografadas provenientes do seu IdP externo, deverá gerar um arquivo de chave privada usando o IdP da sua organização e carregá-lo na configuração SAML do IAM no formato de arquivo .pem. O AWS STS precisa dessa chave privada para descriptografar respostas SAML correspondentes à chave pública que foi carregada para seu IdP.

> **Note**
>
> Conforme definido pela Versão 1.0 do Perfil de Interoperabilidade de Metadados do SAML V2.0, o IAM não avalia nem toma medidas em relação à expiração dos certificados X.509 em documentos de metadados do SAML. Caso não esteja preocupado com certificados X.509 expirados, recomendamos monitorar as datas de expiração dos certificados e alterná-los de acordo com as políticas de governança e segurança da sua organização.

3. No console do IAM, crie um provedor de identidade SAML. Como parte do processo, carregue o documento de metadados SAML e a chave de descriptografia privada que foi gerada pelo IdP na sua organização em Step 2. Para obter mais informações, consulte Criar um provedor de identidades SAML no IAM.
4. No IAM, crie uma ou mais funções do IAM. Na política de confiança da função, defina o provedor do SAML como principal, o que estabelece uma relação de confiança entre sua organização e a AWS. A política de permissões da função estabelece o que os usuários da sua organização têm permissão para fazer na AWS. Para obter mais informações, consulte Criar um perfil para um provedor de identidade de terceiros.

> **Note**
>
> Os IdPs do SAML usados em uma política de confiança de função devem estar na mesma conta em que a função está.

5. No IdP da sua organização, você define asserções que mapeiam usuários ou grupos em sua organização para as funções do IAM. Observe que usuários e grupos diferentes na sua organização podem ser mapeados para funções do IAM distintas. As etapas exatas para executar o mapeamento dependem do IdP que você estiver usando. No cenário anterior de uma pasta do Amazon S3 para usuários, é possível que todos os usuários sejam mapeados para a mesma função que fornece as permissões do Amazon S3. Para obter mais informações, consulte Configurar declarações SAML para a resposta de autenticação.

   Se o seu IdP habilitar o SSO para o console da AWS, configure a duração máxima de sessões do console. Para obter mais informações, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS.

6. No aplicativo que você estiver criando, chame a API AWS Security Token Service do `AssumeRoleWithSAML`, transmitindo para ela o ARN do provedor SAML criado em Step 3, o ARN da função para supor que você criou em Step 4 e a declaração do SAML sobre o usuário atual que você obtém do seu IdP. A AWS garante que a solicitação para assumir a função venha do IdP referenciado no provedor de SAML.

   Para obter mais informações, consulte AssumeRoleWithSAML na Referência da API do AWS Security Token Service.

7. Se a solicitação for bem-sucedida, a API retorna um conjunto de credenciais de segurança temporárias, que o aplicativo pode usar para fazer solicitações assinadas para a AWS. Sua aplicação tem informações sobre o usuário atual e pode acessar pastas específicas do usuário no Amazon S3, conforme descrito no cenário anterior.

Visão geral da função para permitir acesso federado do SAML aos seus recursos da AWS

Os perfis que você cria no IAM definem o que as entidades principais federadas SAML de sua organização têm permissão para fazer na AWS. Ao criar a política de confiança para a função, especifique o provedor SAML criado anteriormente como `Principal`. Além disso, examine a política de confiança com uma `Condition`, para permitir que apenas usuários que satisfaçam determinados atributos do SAML tenham acesso à função. Por exemplo, especifique que apenas usuários cujas afiliações do SAML sejam `staff` (conforme definido em https://openidp.feide.no) tenham permissão para acessar a função, como ilustrado pelo seguinte exemplo de política:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://us-east-1.signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
```

```
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws-cn:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:aud": "https://cn-north-1.signin.amazonaws.cn/saml",
                    "saml:iss": "https://openidp.feide.no"
                },
                "ForAllValues:StringLike": {
                    "saml:edupersonaffiliation": [
                        "staff"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> Os IdPs do SAML usados em uma política de confiança de função devem estar na mesma conta em que a função está.

A chave de contexto `saml:aud` na política especifica o URL exibido no navegador durante o início de sessão no console. Esse URL do endpoint de início de sessão deve corresponder ao atributo de destinatário do seu provedor de identidade. É possível incluir URLs de início de sessão em determinadas regiões. A AWS recomenda usar endpoints regionais em vez do endpoint global para melhorar a resiliência da federação. Se você tiver apenas um endpoint configurado, não poderá se federar na AWS, no improvável caso de o endpoint se tornar indisponível. Para obter uma lista dos possíveis valores de *`region-code`* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS).

O exemplo a seguir mostra o formato do URL de login com a região `region-code` opcional.

`https://`*`region-code`*`.signin.aws.amazon.com/saml`

Se a criptografia SAML for necessária, o URL de início de sessão deverá incluir o identificador exclusivo que a AWS atribui ao seu provedor de SAML, o qual você pode encontrar na página de detalhes do provedor de identidade. No exemplo a seguir, o URL de início de sessão inclui o identificador exclusivo do IdP, que exige que /acs/ seja anexado ao caminho de login.

`https://`*`region-code`*`.signin.aws.amazon.com/saml/acs/`*`IdP-ID`*

Para a política de permissões na função, especifique as permissões como faria para qualquer função. Por exemplo, se os usuários da sua organização tiverem permissão para administrar instâncias do Amazon Elastic Compute Cloud, você deverá permitir explicitamente as ações do Amazon EC2 na política de permissões, como aquelas na política gerenciada AmazonEC2FullAccess.

Para obter mais informações sobre as chaves SAML que você pode verificar em uma política, consulte Chaves disponíveis para federação do AWS STS com base em SAML.

Identificar exclusivamente os usuários na federação baseada em SAML

Quando você cria políticas de acesso no IAM, geralmente é útil poder especificar permissões com base na identidade dos usuários. Por exemplo, para usuários federados usando o SAML, uma aplicação pode manter informações no Amazon S3 usando uma estrutura como esta:

```
amzn-s3-demo-bucket/app1/user1
amzn-s3-demo-bucket/app1/user2
amzn-s3-demo-bucket/app1/user3
```

Você pode criar o bucket (`amzn-s3-demo-bucket`) e a pasta (`app1`) por meio do console do Amazon S3 ou da AWS CLI, uma vez que esses são valores estáticos. No entanto, as pastas

específicas do usuário (*user1*, *user2*, *user3* etc.) precisam ser criadas em tempo de execução usando código, já que o valor que identifica o usuário é desconhecido até a primeira vez que o usuário faz login, por meio do processo de federação.

Para gravar políticas que façam referência a detalhes específicos do usuário como parte de um nome de recurso, a identidade do usuário deve estar disponível nas chaves do SAML que podem ser usadas em condições de política. As seguintes chaves estão disponíveis para federação baseada em SAML 2.0 para uso em políticas do IAM. Você pode usar os valores retornados pelas seguintes chaves para criar identificadores de usuário exclusivos para recursos como pastas do Amazon S3.

- `saml:namequalifier`. Um valor de hash com base na concatenação do valor de resposta de `Issuer` (`saml:iss`) e uma string com o ID da conta da AWS e o nome amigável (a última parte do ARN) do provedor SAML no IAM. A concatenação do ID da conta e do nome amigável do provedor SAML está disponível para as políticas do IAM como a chave `saml:doc`. O ID da conta e o nome do provedor devem ser separados por '/' como em "123456789012/provider_name". Para obter mais informações, consulte a chave do `saml:doc` em Chaves disponíveis para federação do AWS STS com base em SAML.

  A combinação de `NameQualifier` e `Subject` pode ser usada para identificar de forma exclusiva uma entidade principal federada SAML. O pseudocódigo a seguir mostra como esse valor é calculado. Nesse pseudocódigo, + indica concatenação, `SHA1` representa uma função que produz um resumo de mensagens usando SHA-1 e `Base64` representa uma função que produz a versão codificada em Base64 da saída de hash.

  ```
  Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
  MySAMLIdP" ) )
  ```

  Para obter mais informações sobre a chaves de política disponíveis para federação baseada em SAML, consulte Chaves disponíveis para federação do AWS STS com base em SAML.
- `saml:sub` (string). Trata-se do assunto da solicitação, que inclui um valor que identifica de forma exclusiva um usuário individual em uma organização (por exemplo, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (string). Essa chave pode ser `persistent`, `transient` ou o URI `Format` completo dos elementos `Subject` e `NameID` usados na declaração do SAML. Um valor `persistent` indica que o valor em `saml:sub` é o mesmo para um usuário em todas as sessões. Se o valor for `transient`, o usuário terá um valor `saml:sub` diferente para cada sessão. Para obter mais informações sobre o atributo `NameID` do elemento `Format`, consulte Configurar declarações SAML para a resposta de autenticação.

O exemplo a seguir mostra uma política de permissão que usa as chaves anteriores para conceder permissões a uma pasta específica do usuário no Amazon S3. A política supõe que os objetos do Amazon S3 são identificados usando um prefixo que inclui `saml:namequalifier` e `saml:sub`. Observe que o elemento `Condition` inclui um teste para garantir que o `saml:sub_type` esteja definido como `persistent`. Se for definido `transient`, o valor do `saml:sub` para o usuário poderá ser diferente para cada sessão e a combinação de valores não deverá ser usada para identificar pastas específicas do usuário.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
```

```
    ],
    "Resource": [
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}",
      "arn:aws-cn:s3:::amzn-s3-demo-bucket-org-data/backup/${saml:namequalifier}/
${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

Para obter mais informações sobre o mapeamento de declarações do IdP para chaves de política, consulte Configurar declarações SAML para a resposta de autenticação.

Criar um provedor de identidades SAML no IAM

Um provedor de identidade SAML 2.0 do IAM é uma entidade no IAM que descreve um serviço de provedor de identidade (IdP) externo compatível com o padrão SAML 2.0 (Security Assertion Markup Language 2.0). Você usa um provedor de identidade IAM quando deseja estabelecer confiança entre um IdP compatível com SAML, como Shibboleth ou Serviços de Federação do Active Directory e a AWS, para que seus usuários possam acessar recursos da AWS. Os provedores de identidade SAML do IAM são usados como entidades de segurança em uma política de confiança do IAM.

Para ter mais informações sobre esse cenário, consulte Federação SAML 2.0.

Você pode criar e gerenciar um provedor de identidade do IAM no Console de gerenciamento da AWS ou com a AWS CLI, o Tools for Windows PowerShell ou chamadas de API da AWS.

Depois de criar um provedor SAML, você deve criar uma ou mais funções do IAM. Função é uma identidade na AWS que não tem as próprias credenciais (como um usuário). Porém, neste contexto, um perfil é atribuído dinamicamente a uma entidades principal federada que é autenticada pelo seu IdP. O perfil permite que seu IdP solicite credenciais de segurança temporárias para obter acesso à AWS. As políticas atribuídas ao perfil determinam o que os usuários podem fazer na AWS. Para criar uma função para a federação do SAML, consulte Criar um perfil para um provedor de identidade de terceiros.

Finalmente, após criar o perfil, você deve concluir a confiança de SAML configurando o IdP com informações sobre a AWS e os perfis que suas entidades principais federadas deverão usar. Isso é chamado de configurar a confiança da parte dependente entre seu IdP e a AWS. Para configurar

a confiança da parte dependente, consulte Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações.

Tópicos

- Pré-requisitos
- Criar e gerenciar um provedor de identidade SAML do IAM (console)
- Gerenciar chaves de criptografia SAML
- Criar e gerenciar um provedor de identidade SAML do IAM (AWS CLI)
- Criar e gerenciar um provedor de identidade SAML do IAM (API da AWS)
- Próximas etapas

Pré-requisitos

Antes de criar um provedor de identidade SAML, é necessário ter as seguintes informações do IdP.

- Obtenha o documento de metadados SAML do IdP. Esse documento inclui o nome do emissor, informações de expiração e chaves que podem ser usadas para validar a resposta de autenticação SAML (declarações) que são recebidas do IdP. Para gerar o documento de metadados, use o software de gerenciamento de identidade fornecido pelo IdP externo.

  > **Important**
  >
  > Esse arquivo de metadados inclui o nome do emissor, informações de validade e chaves que podem ser usadas para validar a resposta de autenticação do SAML (declarações) que são recebidas do IdP. O arquivo de metadados deve ser codificado no formato UTF-8 sem a marca de ordem de bytes (BOM). Para remover a BOM, você pode codificar o arquivo como UTF-8 usando uma ferramenta de edição de texto, como o Notepad++.
  > O certificado x.509 incluído como parte do documento de metadados do SAML deve usar um tamanho de chave de, pelo menos, 1.024 bits. Além disso, o certificado x.509 também deve estar livre de extensões repetidas. É possível usar extensões, mas elas só podem aparecer uma vez no certificado. Se o certificado x.509 não atender a nenhuma das condições, a criação do IdP vai falhar e retornar um erro “Unable to parse metadata”.
  > Conforme definido pela Versão 1.0 do Perfil de Interoperabilidade de Metadados do SAML V2.0, o IAM não avalia nem toma medidas em relação à expiração dos certificados X.509 em documentos de metadados do SAML. Caso não esteja preocupado com certificados

X.509 expirados, recomendamos monitorar as datas de expiração dos certificados e alterná-los de acordo com as políticas de governança e segurança da sua organização.

- Ao escolher habilitar a criptografia SAML, você deve gerar um arquivo de chave privada usando seu IdP e enviar esse arquivo para sua configuração SAML do IAM no formato de arquivo .pem. O AWS STS precisa dessa chave privada para descriptografar as respostas SAML que correspondem à chave pública enviada para seu IdP. Os seguintes algoritmos são aceitos:
  - Algoritmos de criptografia
    - AES-128
    - AES-256
    - RSA-OAEP
  - Algoritmos de transporte de chaves
    - AES-CBC
    - AES-GCM

  Consulte a documentação do seu provedor de identidade para obter as etapas de geração de uma chave privada.

> **Note**
>
> O IAM Identity Center e o Amazon Cognito não são compatíveis com declarações de SAML criptografadas provenientes de provedores de identidade SAML do IAM. É possível adicionar indiretamente compatibilidade com declarações de SAML criptografadas à federação do banco de identidades do Amazon Cognito com grupos de usuários do Amazon Cognito. Os grupos de usuários têm uma federação SAML que é independente da federação SAML do IAM e é compatível com assinatura e criptografia de SAML. Embora esse atributo não se estenda diretamente aos bancos de identidades, os grupos de usuários podem ser IdPs dos bancos de identidades. Para usar a criptografia SAML com bancos de identidades, adicione um provedor de SAML com criptografia a um grupo de usuários que seja um IdP de um banco de identidades.
> Seu provedor de SAML deverá ser capaz de criptografar declarações de SAML com uma chave fornecida pelo seu grupo de usuários. Os grupos de usuários não aceitarão declarações criptografadas com um certificado fornecido pelo IAM.

Para obter instruções sobre como configurar muitos dos IdPs disponíveis para trabalhar com a AWS, incluindo como gerar o documento de metadados SAML necessários, consulte Integrar provedores de soluções SAML de terceiros com a AWS.

Para obter ajuda com a federação SAML, consulte Solução de problemas com a federação SAML.

Criar e gerenciar um provedor de identidade SAML do IAM (console)

Você pode usar o Console de gerenciamento da AWS para criar, atualizar e excluir provedores de identidade SAML do IAM. Para obter ajuda com a federação SAML, consulte Solução de problemas com a federação SAML.

Para criar um provedor de identidade SAMLdo IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Identity providers (Provedores de identidade) e, em seguida Add provider (Adicionar provedor).
3. Para a opção Configure provider (Configurar provedor), escolha SAML.
4. Digite um nome para o provedor de identidade.
5. Em Metadata document (Documento de metadados), clique em Choose file (Escolher arquivo) e especifique o documento de metadados SAML que você obteve por download no the section called “Pré-requisitos”.

> **Note**
>
> A data Válido até exibida para o provedor de identidade é definida pelo IAM no momento da criação do provedor. Ela não reflete os atributos `validUntil` ou `cacheDuration` em seu documento de metadados SAML. O IAM não usa esses atributos de metadados para determinar a data Válido até.
> O IAM não avalia nem realiza nenhuma ação em relação à expiração do documento de metadados SAML ou aos certificados X.509 em documentos de metadados SAML durante a autenticação SAML. A validade da autenticação SAML é aplicada na declaração, usando o atributo `NotOnOrAfter` nos elementos `Conditions` e `SubjectConfirmationData` da declaração. Caso não esteja preocupado com certificados X.509 expirados, recomendamos monitorar as datas de expiração dos

certificados e alterná-los de acordo com as políticas de governança e segurança da sua organização.

6. (Opcional) Em Criptografia SAML, selecione Escolher arquivo e escolha o arquivo de chave privada que você criou em the section called “Pré-requisitos”. Escolha Exigir criptografia para aceitar somente solicitações criptografadas do IdP.
7. (Opcional) Em Add tags (Adicionar etiquetas), você pode adicionar pares de chave-valor para ajudar na identificação e organização de seus IdPs. Você também pode usar tags para controlar o acesso aos recursos da AWS. Para saber mais sobre a marcação de provedores de identidade SAML, consulte Marcar provedores de identidades SAML do IAM.

   Escolha Adicionar Tag. Insira valores para cada par de chave-valor de tag.
8. Verifique as informações fornecidas. Quando terminar, escolha Add provider (Adicionar provedor).
9. Atribua um perfil do IAM ao seu provedor de identidade. Esse perfil fornece a identidades de usuário externas gerenciadas pelo seu provedor de identidade permissões para acessar recursos da AWS em sua conta. Para saber mais sobre como criar funções para a federação de identidades, consulte Criar um perfil para um provedor de identidade de terceiros.

> **Note**
>
> Os IdPs do SAML usados em uma política de confiança de função devem estar na mesma conta em que a função está.

Para excluir um provedor SAML (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Identity providers (Provedores de identidade).
3. Marque a caixa de seleção ao lado do provedor de identidade que você deseja excluir.
4. Escolha Excluir. Uma nova janela é aberta.
5. Confirme se deseja excluir o provedor digitando a palavra `delete` no campo. Em seguida, selecione Excluir.

Gerenciar chaves de criptografia SAML

Você pode configurar provedores SAML do IAM para receber declarações criptografadas na resposta SAML do IdP externo. Os usuários podem assumir um perfil AWS com declarações SAML criptografadas chamando `sts:AssumeRoleWithSAML`.

A criptografia SAML garante que as declarações sejam seguras quando passadas por intermediários ou terceiros. Além disso, esse atributo ajuda a manter a conformidade com o FedRAMP ou quaisquer requisitos de política de conformidade interna que exijam a criptografia de declarações SAML.

Para configurar um provedor de identidade SAML do IAM, consulte Criar um provedor de identidades SAML no IAM. Para obter ajuda com a federação SAML, consulte Solução de problemas com a federação SAML.

Alternar a chave de criptografia SAML

O IAM usa a chave privada que você enviou ao provedor SAML do IAM para descriptografar declarações SAML criptografadas do seu IdP. É possível salvar até dois arquivos de chave privada para cada provedor de identidade, o que permite a alternância entre as chaves privadas conforme necessário. Quando dois arquivos são salvos, cada solicitação tentará primeiro descriptografar com a data de Adição mais recente e, em seguida, o IAM tentará descriptografar a solicitação com a data de Adição mais antiga.

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Provedores de identidade e, em seguida, selecione seu provedor na lista.
3. Escolha a guia Criptografia SAML e depois Adicionar nova chave.
4. Selecione Escolher arquivo e carregue a chave privada que você baixou do IdP como um arquivo .pem. Em seguida, escolha Adicionar chave.
5. Na seção Chaves privadas para descriptografia SAML, selecione o arquivo de chave privada expirado e escolha Remover. Recomendamos remover a chave privada expirada após adicionar uma nova chave privada para garantir que a primeira tentativa de descriptografar sua declaração seja bem-sucedida.

Criar e gerenciar um provedor de identidade SAML do IAM (AWS CLI)

Você pode usar a AWS CLI para criar, atualizar e excluir provedores SAML. Para obter ajuda com a federação SAML, consulte Solução de problemas com a federação SAML.

Para criar um provedor de identidade do IAM e carregar um documento de metadados (AWS CLI)

- Execute este comando: `aws iam create-saml-provider`

Para atualizar um provedor de identidade SAML do IAM (AWS CLI)

É possível atualizar o arquivo de metadados e as configurações de criptografia SAML, bem como alternar arquivos de descriptografia de chave privada para seu provedor SAML do IAM. Para alternar chaves privadas, adicione sua nova chave privada e, em seguida, remova a chave antiga em uma solicitação separada. Para obter mais informações sobre a alternância de chaves privadas, consulte Gerenciar chaves de criptografia SAML.

- Execute este comando:`aws iam update-saml-provider`

Para etiquetar um provedor de identidade do IAM existente (AWS CLI)

- Execute este comando:`aws iam tag-saml-provider`

Para listar etiquetas para o provedor de identidade do IAM existente (AWS CLI)

- Execute este comando:`aws iam list-saml-provider-tags`

Para remover etiquetas de um provedor de identidade do IAM existente (AWS CLI)

- Execute este comando:`aws iam untag-saml-provider`

Para excluir um provedor de identidade SAML do IAM (AWS CLI)

1. (Opcional) Para listar informações para todos os provedores, como o ARN, data de criação e expiração, execute o seguinte comando:
   - `aws iam list-saml-providers`

2. (Opcional) Para obter informações sobre um provedor específico, como ARN, data de criação, data de expiração, configurações de criptografia e informações de chave privada, execute o seguinte comando:

   - `aws iam get-saml-provider`

3. Para excluir um provedor de identidade do IAM, execute o seguinte comando:

   - `aws iam delete-saml-provider`

Criar e gerenciar um provedor de identidade SAML do IAM (API da AWS)

Você pode usar a API da AWS para criar, atualizar e excluir provedores SAML. Para obter ajuda com a federação SAML, consulte Solução de problemas com a federação SAML.

Para criar um provedor de identidadedo IAM e carregar um documento de metadados (API da AWS)

- Chame esta operação: `CreateSAMLProvider`

Para atualizar um provedor de identidade SAML do IAM (API da AWS)

É possível atualizar o arquivo de metadados e as configurações de criptografia SAML, bem como alternar arquivos de descriptografia de chave privada para seu provedor SAML do IAM. Para alternar chaves privadas, adicione sua nova chave privada e, em seguida, remova a chave antiga em uma solicitação separada. Para obter mais informações sobre a alternância de chaves privadas, consulte Gerenciar chaves de criptografia SAML.

- Chame esta operação: `UpdateSAMLProvider`

Para etiquetar um provedor de identidade existente do IAM (API da AWS)

- Chame esta operação: `TagSAMLProvider`

Para listar etiquetas para um provedor de identidade do IAM existente (API da AWS)

- Chame esta operação: `ListSAMLProviderTags`

Para remover etiquetas em um provedor de identidade existente do IAM (API da AWS)

- Chame esta operação: `UntagSAMLProvider`

Para excluir um provedor de identidade do IAM (API da AWS)

1. (Opcional) Para listar informações para todos os IdPs, como o ARN, data de criação e expiração, chame a seguinte operação:
   - `ListSAMLProviders`
2. (Opcional) Para obter informações sobre um provedor específico, como ARN, data de criação, data de expiração, configurações de criptografia e informações de chave privada, chame a seguinte operação:
   - `GetSAMLProvider`
3. Para excluir um IdP, chame a seguinte operação:
   - `DeleteSAMLProvider`

Próximas etapas

Depois de criar um provedor de identidade do SAML, configure a confiança da parte confiável com o seu IdP. Também é possível usar declarações da resposta de autenticação do seu IdP em políticas para controlar o acesso a um perfil.

- Você deve informar ao IdP sobre a AWS como um provedor de serviços. Isso é chamado de adicionar confiança da parte confiável entre seu IdP e a AWS. O processo exato de adicionar a confiança da parte dependente de qual IdP você está usando. Para obter detalhes, consulte Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações.
- Quando o IdP envia a resposta contendo as solicitações para a AWS, muitas das solicitações de entrada são mapeadas para as chaves de contexto da AWS. Você pode utilizar essas chaves de contexto nas políticas do IAM usando o elemento Condição para controlar o acesso a um perfil. Para obter mais detalhes, consulte Configurar declarações SAML para a resposta de autenticação

Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações

Quando você cria um provedor de identidade do IAM e a função para acesso SAML, você está informando a AWS sobre o provedor de identidade (IdP) externo e o que seus usuários podem fazer. Em seguida, a sua próxima etapa é informar o IdP sobre a AWS como provedor de serviços. Isso é chamado de adicionar confiança da parte dependente entre seu IdP e a AWS. O processo exato de adicionar a confiança da parte dependente de qual IdP você está usando. Para ver detalhes, consulte a documentação do seu software de gerenciamento de identidade.

Muitos IdPs permitem que você especifique uma URL a partir do qual o IdP lerá um documento XML contendo informações e certificados da parte dependente. Para AWS, use o URL do endpoint de início de sessão. O exemplo a seguir mostra o formato do URL com a região `region-code` opcional.

`https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

Se a criptografia SAML for necessária, o URL deverá incluir o identificador exclusivo atribuído pela AWS ao seu provedor SAML, o qual você pode encontrar na página de detalhes do provedor de identidade. O exemplo a seguir mostra um URL de início de sessão regional que inclui um identificador exclusivo.

`https://region-code.signin.aws.amazon.com/static/saml/IdP-ID/saml-metadata.xml`

Para obter uma lista dos possíveis valores de *`region-code`* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS). Para o valor AWS, também é possível usar o endpoint não regional `https://signin.aws.amazon.com/saml`.

Se você não especificar uma URL diretamente, faça download do documento XML da URL anterior e importe-o para o seu software de IdP.

Você também precisa criar regras de reivindicação apropriadas no seu IdP que especifiquem a AWS como uma parte dependente. Quando o IdP envia uma resposta SAML ao endpoint da AWS, ele inclui uma declaração SAML que contém uma ou mais reivindicações. Reivindicação são informações sobre o usuário e seus grupos. Uma regra de solicitação mapeia essas informações aos atributos SAML. Isso permite que você verifique se as respostas de autenticação SAML do seu IdP contêm os atributos necessários que a AWS usa em políticas do IAM para verificar as permissões para entidades principais federadas SAML. Para saber mais, consulte os seguintes tópicos:

- Visão geral da função para permitir acesso federado do SAML aos seus recursos da AWS. Este tópico discute o uso de chaves específicas do SAML em políticas do IAM e como usá-las para restringir permissões para entidades principais federadas SAML.
- Configurar declarações SAML para a resposta de autenticação. Este tópico discute como configurar declarações SAML que incluem informações sobre o usuário. As solicitações são fornecidas em uma declaração SAML e incluídas na resposta SAML que é enviada à AWS. Você deve garantir que as informações necessárias às políticas da AWS sejam incluídas na declaração SAML em um formato que a AWS possa reconhecer e usar.
- Integrar provedores de soluções SAML de terceiros com a AWS. Este tópico fornece links para a documentação fornecida por outras organizações sobre como integrar soluções de identidade com a AWS.

> **Note**
>
> Para melhorar a resiliência da federação, recomendamos que você configure seu IdP e sua federação da AWS para oferecer suporte a vários endpoints de login do SAML. Para obter detalhes, consulte o artigo do AWS Security Blog How to use regional SAML endpoints for failover.

Integrar provedores de soluções SAML de terceiros com a AWS

> **Note**
>
> Recomendamos exigir que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Você já pensou em usar o Centro de Identidade do AWS IAM? O IAM Identity Center pode ser usado para gerenciar centralmente o acesso a várias Contas da AWS e fornecer aos usuários acesso de logon único protegido por MFA a todas as contas atribuídas em um só lugar. Com o IAM Identity Center, é possível criar e gerenciar identidades de usuários no IAM Identity Center ou conectar facilmente ao provedor de identidades compatível com SAML 2.0 existente. Para obter mais informações, consulte O que é o IAM Identity Center? no Guia do usuário do Centro de Identidade do AWS IAM.

Os links a seguir ajudam a configurar soluções de provedor de identidades (IdP) SAML 2.0 de terceiros para trabalhar com as federações do AWS. Consulte seu provedor de identidade para

determinar se ele oferece suporte à criptografia de tokens SAML. Para conhecer os requisitos de criptografia SAML, consulte Gerenciar chaves de criptografia SAML.

> **Tip**
>
> AWSOs engenheiros do Support podem ajudar os clientes que tenham planos de suporte Business e Enterprise em algumas tarefas de integração que envolvem software de terceiros. Para obter uma lista atual de plataformas e aplicativos compatíveis, consulte Quais softwares de terceiros são compatíveis? nas Perguntas frequentes do AWS Support.

Solução	Mais informações
Auth0	Integrate with Amazon Web services (Integrar com Amazon Web Services): esta página no site de documentação do Auth0 tem links para recursos que descrevem como configurar a autenticação única (SSO) com o Console de gerenciamento da AWS e inclui um exemplo de JavaScrip t. Você pode configurar Auth0 para passar tags de sessão. Para obter mais informações, consulte Auth0 Announces Partnership with AWS for IAM Session Tags.
Microsoft Entra	Tutorial: Integração do SSO do Microsoft Entra ao AWS Single-Account Access: este tutorial no site da Microsoft descreve como configurar o Microsoft Entra (anterior mente conhecido como Azure AD) como um provedor de identidades (IdP) usando a federação SAML.
Centrify	Configure Centrify and Use SAML for SSO to AWS: esta página no site da Centrify explica como configurar o Centrify para usar SAML para SSO na AWS.
CyberArk	Configure o CyberArk para fornecer acesso à Amazon Web Services (AWS) aos usuários que fazem login por meio de autenticação única (SSO) do SAML pelo CyberArk User Portal.

Solução	Mais informações
ForgeRock	A Plataforma de identidade ForgeRock integra-se à AWS. É possível configurar a ForgeRock para transmitir tags de sessão. Para obter mais informações, consulte Attribute Based Access Control for Amazon Web Services.
Google Workspace	Aplicação na nuvem da Amazon Web Services: este artigo no site de ajuda da administração do Google Workspace descreve como configurar o Google Workspace como um IdP SAML 2.0 utilizando a AWS como provedor de serviços.
IBM	Você pode configurar o IBM para passar tags de sessão. Para obter mais informações, consulte IBM Cloud Identity IDaaS one of first to support AWS session tags.
JumpCloud	Conceder acesso por meio de perfis do IAM para autenticação única (SSO) com a Amazon AWS: este artigo no site da JumpCloud descreve como configurar e habilitar o SSO com base em perfis do IAM para a AWS.
Matrix42	Guia de conceitos básicos do MyWorkspace: este guia descreve como integrar os serviços de identidade da AWS com o Matrix42 MyWorkspace.

Solução	Mais informações
Microsoft Active Directory Federation Services (AD FS)	Notas de campo: integração do Active Directory Federation Service com o Centro de Identidade do AWS IAM: esta publicação no blog AWS Architecture explica o fluxo de autenticação entre o AD FS e o Centro de Identidade do AWS IAM (Centro de Identidade do IAM). O IAM Identity Center oferece suporte à federação de identidades com SAML 2.0, permitindo a integração com soluções do AD FS. Os usuários podem fazer login no portal do IAM Identity Center com suas credenciais corporativas reduzindo a sobrecarga administrativa de manter credenciais separadas no IAM Identity Center. Você também pode configurar o AD FS para passar tags de sessão. Para obter mais informações, consulte Use attribute-based access control with AD FS to simplify IAM permissions management.
miniOrange	SSO para a AWS: esta página no site do miniOrange descreve como estabelecer acesso seguro à AWS para empresas e controle total sobre o acesso de aplicações da AWS.
Okta	Integrar a interface da linha de comando da Amazon Web Services com o Okta: nesta página no site de suporte do Okta, você pode aprender a configurar o Okta para uso com a AWS. Você pode configurar o Okta para passar tags de sessão. Para obter mais informações, consulte Okta and AWS Partner to Simplify Access Via Session Tags.
Okta	Federação de contas da AWS: esta seção no site da Okta descreve como configurar e habilitar o Centro de Identidade do IAM para a AWS.

Solução	Mais informações
OneLogin	No OneLogin Knowledgebase, procure **SAML AWS** para obter uma lista de artigos que explicam como configurar a funcionalidade do IAM Identity Center entre o OneLogin e a AWS para cenários de um ou vários perfis. Você pode configurar o OneLogin para passar tags de sessão. Para obter mais informações, consulte OneLogin and Session Tags: Attribute-Based Access Control for AWS Resources.
Ping Identity	Conector PingFederate da AWS: visualize detalhes sobre o conector PingFederate da AWS, um modelo de conexão rápida para configurar facilmente uma autenticação única (SSO) e uma conexão de provisionamento. Leia a documentação e faça download do conector PingFeder ate da AWS mais recente para integrações com a AWS. Você pode configurar o Ping Identity para passar tags de sessão. Para obter mais informações, consulte Announcing Ping Identity Support for Attribute-Based Access Control in AWS.
RadiantLogic	Parceiros de tecnologia da Radiant Logic: o serviço de identidade federada RadiantOne da Radiant Logic se integra à AWS para fornecer um hub de identidade para SSO baseada em SAML.
RSA	O Amazon Web Services - RSA Ready Implementation Guide fornece orientação para integração da AWS e do RSA. Para obter mais informações sobre a configuração do SAML, consulte Amazon Web Services - SAML My Page SSO Configuration - RSA Ready Implementation Guide.
Salesforce.com	Como configurar SSO do Salesforce para a AWS: este artigo de instruções sobre o site do desenvolvedor do Salesforce.com descreve como configurar um provedor de identidade (IdP) no Salesforce e configurar a AWS como um provedor de serviços.

Solução	Mais informações
SecureAuth	AWS - SecureAuth SAML SSO: este artigo no site do SecureAuth descreve como configurar a integração do SAML com a AWS para um dispositivo SecureAuth.
Shibboleth	How to Use Shibboleth for SSO to the Console de gerenciamento da AWS: este registro no AWS Security Blog fornece um tutorial detalhado sobre como instalar o Shibboleth e configurá-lo como provedor de identidad e para a AWS. Você pode configurar o Shibboleth para passar tags de sessão.

Para obter mais detalhes, consulte a página Parceiros do IAM no site da AWS.

Configurar declarações SAML para a resposta de autenticação

Após a verificação da identidade de um usuário na sua organização, o provedor de identidades (IdP) externo envia uma resposta de autenticação ao URL do endpoint de início de sessão da AWS. Essa resposta é uma solicitação POST que inclui um token SAML que segue o padrão Vinculação POST de HTTP para SAML 2.0 e contém os seguintes elementos ou solicitações. Você configura essas solicitações em seu IdP compatível com SAML. Consulte a documentação de seu IdP para obter instruções sobre como inserir essas solicitações.

Quando o IdP envia a resposta contendo as solicitações para a AWS, muitas das solicitações de entrada são mapeadas para as chaves de contexto da AWS. Essas chaves de contexto podem ser verificadas nas políticas do IAM usando o elemento `Condition`. Há uma lista de mapeamentos disponíveis na seção Mapeamento de atributos SAML para chaves de contexto de política de confiança da AWS.

`Subject` e da **`NameID`**

A resposta deve incluir exatamente um elemento `SubjectConfirmation` com um elemento `SubjectConfirmationData` que inclui ambos os atributos `NotOnOrAfter` e `Recipient`. O atributo Recipient deve incluir um valor que corresponda ao URL do endpoint de início de sessão da AWS. Seu IdP pode usar o termo `ACS`, `Recipient` ou `Target` para fazer referência a esse atributo.

Se a criptografia SAML for necessária, o URL de início de sessão deverá incluir o identificador exclusivo que a AWS atribui ao seu provedor de SAML, o qual você pode encontrar na página de

detalhes do provedor de identidade. O exemplo a seguir mostra o formato do URL de login com a região `region-code` opcional.

`https://region-code.signin.aws.amazon.com/saml`

No exemplo a seguir, o URL de início de sessão inclui um identificador exclusivo que exige que /acs/ seja anexado ao caminho de início de sessão.

`https://region-code.signin.aws.amazon.com/saml/acs/IdP-ID`

Para obter uma lista dos possíveis valores de *region-code* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS). Para o valor AWS, também é possível usar o endpoint de sign-in global `https://signin.aws.amazon.com/saml`.

Os elementos `NameID` podem ter o valor persistente, transitório ou consistir no URI de formato completo, conforme fornecido pela solução IdP. O valor persistente indica que o valor em `NameID` é o mesmo para um usuário entre as sessões. Se o valor for transitório, o usuário terá um valor `NameID` diferente para cada sessão. As interações de login único oferecem suporte aos seguintes tipos de identificadores:

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

O trecho a seguir mostra um exemplo. Substitua os valores marcados pelos seus próprios valores.

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://region-code.signin.aws.amazon.com/saml/SAMLSP4SHN3UIS2D558H46"/>
```

```
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> A chave de contexto `saml:aud` vem do atributo destinatário SAML porque é o equivalente SAML ao campo de público do OIDC, por exemplo, `accounts.google.com:aud`.

`PrincipalTag`Atributo SAML

(Opcional) Você pode usar um elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`*. Este elemento permite que você passe atributos como tags de sessão na declaração do SAML. Para obter mais informações sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

Para passar atributos como tags de sessão, inclua o elemento `AttributeValue` que especifica o valor da tag. Por exemplo, para passar os pares chave-valor de tag `Project` = `Marketing` e `CostCenter` = `12345`, use o atributo a seguir. Inclua um elemento `Attribute` separado para cada tag.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Para definir as tags acima como transitivas, inclua outro elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Ele é um atributo multivalor opcional que define suas tags de sessão como transitivas. As tags transitivas persistem quando você usa a sessão do SAML para assumir outra função na AWS. Isso é conhecido como encadeamento de funções. Por exemplo, para definir as tags `Principal` e `CostCenter` como transitivas, use o atributo a seguir para especificar as chaves.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

RoleAtributo SAML

Você pode usar um elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/Role` Esse elemento contém um ou mais elementos `AttributeValue` que listam o provedor de identidade do IAM e a função para a qual o usuário é mapeado pelo IdP. A função e o provedor de identidade do IAM são especificados como um par de ARNs delimitado por vírgulas no mesmo formato que os parâmetros `RoleArn` e `PrincipalArn` que são passados para AssumeRoleWithSAML. Esse elemento deve conter pelo menos um par de provedor e função (elemento `AttributeValue`) e pode conter vários pares. Se o elemento contiver vários pares, será solicitado que o usuário escolha qual perfil assumir quando usar o WebSSO para fazer login no Console de gerenciamento da AWS.

> **Important**
>
> O valor do atributo `Name` na tag `Attribute` faz distinção de maiúsculas e minúsculas. Ele deve ser definido exatamente como `https://aws.amazon.com/SAML/Attributes/Role`.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

RoleSessionNameAtributo SAML

Você pode usar um elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/RoleSessionName` Este elemento contém um elemento `AttributeValue` que fornece um identificador para as credenciais temporárias que são emitidas quando a função é assumida. Você pode usar isso para associar as credenciais temporárias ao usuário que está usando sua aplicação. Esse elemento é usado para exibir as informações do usuário no Console de gerenciamento da AWS. O valor no elemento `AttributeValue` deve ter entre 2 e 64 caracteres, pode conter apenas caracteres alfanuméricos, sublinhados e os seguintes caracteres: . , + = @ - (hífen). Ele nome não pode conter espaços. O valor é geralmente um ID de

usuário (`john`) ou um endereço de e-mail (`johndoe@example.com`). Ele não deve ser um valor que inclua espaço, como o nome de exibição de um usuário (`John Doe`).

> **Important**
>
> O valor do atributo `Name` na tag `Attribute` faz distinção de maiúsculas e minúsculas. Ele deve ser definido exatamente como `https://aws.amazon.com/SAML/Attributes/RoleSessionName`.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

`SessionDuration`Atributo SAML

(Opcional) Você pode usar um elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/SessionDuration`. Esse elemento contém um elemento `AttributeValue` que especifica por quanto tempo o usuário pode acessar o Console de gerenciamento da AWS antes de precisar solicitar novas credenciais temporárias. O valor é um número inteiro que representa o número de segundos da sessão. O valor pode variar de 900 segundos (15 minutos) a 43.200 segundos (12 horas). Se esse atributo não estiver presente, a credencial será válida por uma hora (o valor padrão do parâmetro `DurationSeconds` da API `AssumeRoleWithSAML`).

Para usar esse atributo, você deve configurar o provedor SAML para fornecer acesso de logon único ao Console de gerenciamento da AWS pelo endpoint da web de entrada do console em `https://region-code.signin.aws.amazon.com/saml`. Para obter uma lista dos possíveis valores de *`region-code`* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS). Ou você pode usar o seguinte URL: `https://signin.aws.amazon.com/static/saml`. Observe que esse atributo estende sessões apenas para o Console de gerenciamento da AWS. Ele não pode ampliar a vida útil de outras credenciais. No entanto, se estiver presente em uma chamada de API `AssumeRoleWithSAML`, ele poderá ser usado para abreviar a duração da sessão. O tempo útil padrão das credenciais retornadas pela chamada é de 60 minutos.

Observe também que, se um atributo `SessionNotOnOrAfter` também estiver definido, o menor valor de dois atributos, `SessionDuration` ou `SessionNotOnOrAfter`, estabelecerá a duração máxima da sessão do console.

Quando você ativa as seções do console com uma duração estendida, o risco de comprometimento das credenciais aumenta. Para ajudar a reduzir esse risco, você pode desabilitar imediatamente as sessões ativas do console para qualquer função escolhendo a opção Revoke Sessions (Revogar sessões) na página Role Summary (Resumo da função) no console do IAM. Para obter mais informações, consulte Revogar as credenciais de segurança temporárias do perfil do IAM.

> **Important**
>
> O valor do atributo `Name` na tag `Attribute` faz distinção de maiúsculas e minúsculas. Ele deve ser definido exatamente como `https://aws.amazon.com/SAML/Attributes/SessionDuration`.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

`SourceIdentity`Atributo SAML

(Opcional) Você pode usar um elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Este elemento contém um elemento `AttributeValue` que fornece um identificador da pessoa ou da aplicação que está usando uma função do IAM. O valor da identidade-fonte persiste quando você usa a sessão do SAML para assumir outra função na AWS conhecida como encadeamento de funções. O valor da identidade-fonte está presente na solicitação para cada ação executada durante a sessão da função. O valor definido não pode ser alterado durante a sessão da função. Os administradores podem então usar logs do AWS CloudTrail para monitorar e auditar as informações de identidade-fonte para determinar quem executou ações com funções compartilhadas.

O valor no elemento `AttributeValue` deve ter entre 2 e 256 caracteres, pode conter apenas caracteres alfanuméricos, sublinhados e os seguintes caracteres: . , + = @ - (hífen). Ele nome não pode conter espaços. O valor geralmente é um atributo associado ao usuário, como um id de usuário (`john`) ou um endereço de e-mail (`johndoe@example.com`). Ele não deve ser um valor que inclua espaço, como o nome de exibição de um usuário (`John Doe`). Para obter mais informações

sobre como usar a identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

> **Important**
>
> Se sua asserção SAML estiver configurada para usar o atributo `SourceIdentity`, sua política de confiança da função também deverá incluir a ação `sts:SetSourceIdentity`, caso contrário, a operação de assumir função falhará. Para obter mais informações sobre como usar a identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

Para passar um atributo de identidade-fonte, inclua o elemento `AttributeValue` que especifica o valor da identidade-fonte. Por exemplo, para passar a identidade-fonte `Diego` use o atributo a seguir.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>Diego</AttributeValue>
```

Mapeamento de atributos SAML para chaves de contexto de política de confiança da AWS

As tabelas desta seção listam atributos SAML utilizados com frequência e como eles são mapeados para chaves de contexto de condições de política de confiança na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para isso, compare as chaves com os valores incluídos nas declarações que acompanham uma solicitação de acesso SAML.

> **Important**
>
> Essas chaves estão disponíveis apenas em políticas de confiança do IAM (políticas que determinam quem pode assumir uma função) e não são aplicáveis a políticas de permissões.

Na tabela de atributos eduPerson e eduOrg, os valores são digitados como strings ou listas de strings. Para valores de string, você pode testar esses valores em políticas de confiança do IAM usando condições `StringEquals` ou `StringLike`. Para valores que contêm uma lista de strings, você pode usar os `ForAnyValue`operadores definidos de políticaForAllValues e para testar os valores nas políticas de confiança.

> **Note**
>
> Você deve incluir apenas uma solicitação por chave de contexto da AWS. Se você incluir mais de uma, apenas uma delas será mapeada.

A tabela a seguir mostra os atributos eduPerson e eduOrg.

Atributo eduPerson ou eduOrg (chave **`Name`**)	É mapeado para essa chave de contexto da AWS (chave **`FriendlyName`**)	Tipo
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPersonAffiliation`	Lista de strings
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Lista de strings
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPersonOrgUnitDN`	Lista de strings
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPersonPrimaryAffiliation`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPersonPrincipalName`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPersonEntitlement`	Lista de strings
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPersonPrimaryOrgUnitDN`	String
`urn:oid:1.3.6.1.4.1.5923.1.1.1.9`	`eduPersonScopedAffiliation`	Lista de strings

Atributo eduPerson ou eduOrg (chave **Name**)	É mapeado para essa chave de contexto da AWS (chave **FriendlyName**)	Tipo
urn:oid:1.3.6.1.4.1.5923.1.1.1.10	eduPersonTargetedID	Lista de strings
urn:oid:1.3.6.1.4.1.5923.1.1.1.11	eduPersonAssurance	Lista de strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.2	eduOrgHomePageURI	Lista de strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.3	eduOrgIdentityAuthNPolicyURI	Lista de strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.4	eduOrgLegalName	Lista de strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.5	eduOrgSuperiorURI	Lista de strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.6	eduOrgWhitePagesURI	Lista de strings
urn:oid:2.5.4.3	cn	Lista de strings

A tabela a seguir mostra atributos do Active Directory.

Atributo do AD	É mapeado para essa chave de contexto da AWS	Tipo
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name	name	String
http://schemas.xmlsoap.org/claims/CommonName	commonName	String

Atributo do AD	É mapeado para essa chave de contexto da AWS	Tipo
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	String
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	String
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	String

A tabela a seguir mostra atributos do X.500.

Atributo X.500	É mapeado para essa chave de contexto da AWS	Tipo
`2.5.4.3`	`commonName`	String
`2.5.4.4`	`surname`	String
`2.4.5.42`	`givenName`	String
`2.5.4.45`	`x500UniqueIdentifier`	String
`0.9.2342.19200300100.1.1`	`uid`	String
`0.9.2342.19200300100.1.3`	`mail`	String
`0.9.2342.19200300.100.1.45`	`organizationStatus`	String

Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS

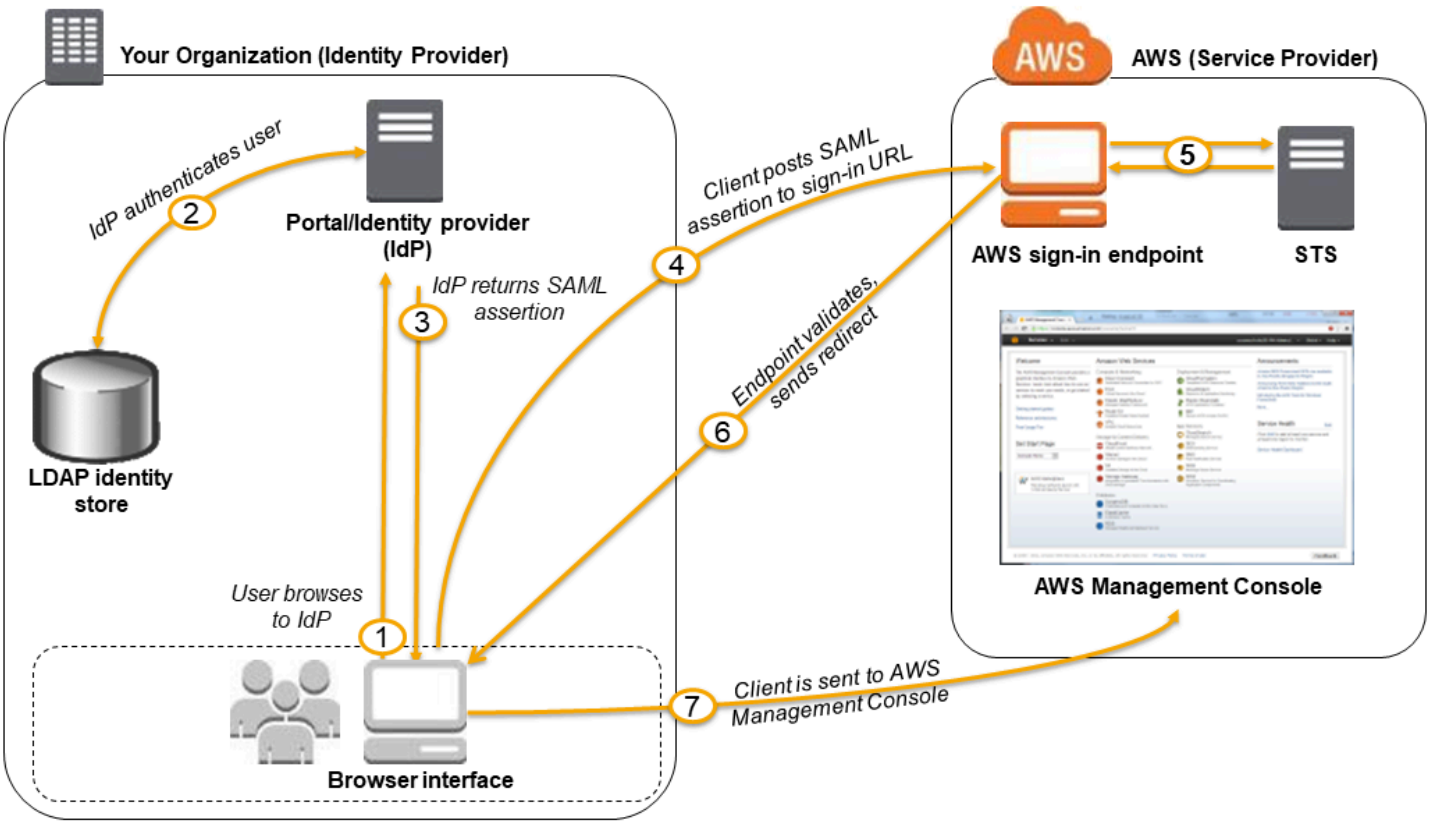
É possível usar um perfil para configurar seu provedor de identidades (IdP) compatível com SAML 2.0 e o AWS para permitir que entidades principais federadas acessem o Console de gerenciamento da AWS. A função concede ao usuário permissões para executar tarefas no console. Se você deseja oferecer às entidades principais federadas SAML outras maneiras de acessar a AWS, consulte um dos seguintes tópicos:

- AWS CLI: Alterar para uma perfil do IAM (AWS CLI)
- Tools for Windows PowerShell: Alternar para um perfil do IAM (Ferramentas para Windows PowerShell)
- AWS API: Alternar para um perfil do IAM (API da AWS)

Visão geral

O diagrama a seguir ilustra o fluxo para logon único habilitado para o SAML.

> **Note**
>
> Esse uso específico de SAML difere do uso mais geral ilustrado em Federação SAML 2.0, pois esse um fluxo de trabalho abre o Console de gerenciamento da AWS em nome do usuário. Isso requer o uso do endpoint de login da AWS em vez de chamar diretamente a API `AssumeRoleWithSAML`. O endpoint chama a API para o usuário e retorna uma URL que redireciona automaticamente o navegador do usuário para o Console de gerenciamento da AWS.



O diagrama ilustra as seguintes etapas:

1. O usuário acessa o portal de sua organização e seleciona a opção de ir para o Console de gerenciamento da AWS. Em sua organização, o portal geralmente é uma função do seu IdP que lida com a troca de confiança entre a organização e a AWS. Por exemplo, no Active Directory Federation Services, o portal URL é: `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`
2. O portal verifica a identidade do usuário na sua organização.
3. O portal gera uma resposta de autenticação SAML que inclui declarações que identificam o usuário e incluem atributos sobre o usuário. Você também pode configurar seu IdP para incluir um atributo de declaração SAML chamado `SessionDuration` que especifica quanto tempo a sessão do console é válida. Você também pode configurar o IdP para passar atributos como tags de sessão. O portal envia essa resposta ao navegador cliente.
4. O navegador cliente é redirecionado para o endpoint de logon único da AWS e publica a declaração SAML.
5. O endpoint solicita credenciais de segurança temporárias em nome do usuário e cria um URL de login do console que usa essas credenciais.

6. AWSA envia o URL de login de volta para o cliente como um redirecionamento.
7. O navegador cliente é redirecionado para o Console de gerenciamento da AWS. Se a resposta de autenticação SAML incluir atributos que sejam mapeados para várias funções do IAM, será solicitado que o usuário selecione a função para acessar o console.

Do ponto de vista do usuário, o processo ocorre de maneira transparente: o usuário começa no portal interno de sua organização e termina no Console de gerenciamento da AWS, sem nunca precisar fornecer nenhuma credencial da AWS.

Consulte as seções a seguir para obter uma visão geral de como configurar esse comportamento junto com links para ver as etapas detalhadas.

Configurar sua rede como um provedor SAML para a AWS

Na rede de sua organização, configure o armazenamento de identidades (como Windows Active Directory) para trabalhar com um IdP baseado em SAML como Windows Active Directory Federation Services, Shibboleth, etc. Usando seu IdP, gere o documento de metadados que descreve sua organização como um IdP e inclui chaves de autenticação. Você também pode configurar o portal da sua organização para rotear as solicitações do usuário do Console de gerenciamento da AWS para o endpoint SAML da AWS para autenticação usando declarações SAML. Como você configura seu IdP para produzir o arquivo metadata.xml depende do IdP. Consulte a documentação do IdP para obter instruções, ou consulte Integrar provedores de soluções SAML de terceiros com a AWS para obter os links para a documentação da web para muitos dos provedores SAML suportados.

Criar um provedor SAML no IAM

Em seguida, faça login no Console de gerenciamento da AWS e vá para o console do IAM. La você cria um novo provedor SAML, que é uma entidade no IAM que mantém informações sobre o IdP de sua organização. Como parte do processo, carregue o documento de metadados produzido pelo software de IdP em sua empresa mencionado na seção anterior. Para obter detalhes, consulte Criar um provedor de identidades SAML no IAM.

Configurar permissões no AWS para entidades principais federadas SAML

A próxima etapa é criar uma função do IAM que defina uma relação de confiança entre o IAM e o IdP da sua organização. Essa função deve identificar o IdP como um principal (entidade confiável) para fins de federação. Essa função também define o que os usuários autenticados pelo IdP de sua organização têm permissão para fazer na AWS. Você pode usar o console do IAM para criar

essa função. Ao criar a política de confiança que indica quem pode assumir a função, especifique o provedor SAML criado anteriormente no IAM. Especifique também um ou mais atributos SAML que o usuário deve atender para poder assumir a função. Por exemplo, você pode especificar que apenas usuários cujo valor [eduPersonOrgDN](#) SAML seja `ExampleOrg` tenham permissão para fazer login. O assistente de função adiciona automaticamente uma condição para testar o atributo `saml:aud` para garantir que a função seja assumida apenas para login no Console de gerenciamento da AWS.

Se a criptografia SAML for necessária, o URL de início de sessão deverá incluir o identificador exclusivo que a AWS atribui ao seu provedor de SAML, o qual você pode encontrar na página de detalhes do provedor de identidade. A política de confiança da função pode ter a seguinte aparência:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/
ExampleOrgSSOProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:edupersonorgdn": "ExampleOrg",
                    "saml:aud": "https://region-code.signin.aws.amazon.com/saml/
acs/SAMLSP4SHN3UIS2D558H46"
                }
            }
        }
    ]
}
```

> **Note**
>
> Os IdPs do SAML usados em uma política de confiança de função devem estar na mesma conta em que a função está.

Recomendamos o uso de endpoints regionais para o atributo `saml:aud` em `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Para obter uma lista dos possíveis valores de *region-code* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS).

Para a política de permissões na função, você especifica permissões como faria para qualquer função, usuário ou grupo. Por exemplo, se os usuários da sua organização tiverem permissão para administrar instâncias do Amazon EC2, você poderá permitir explicitamente as ações do Amazon EC2 na política de permissões. Você pode fazer isso atribuindo uma política gerenciada, como a política gerenciada Acesso total ao Amazon EC2.

Para obter detalhes sobre a criação de uma função para um IdP SAML, consulte Criar um perfil para uma federação do SAML 2.0 (console).

Concluir configuração e criar declarações SAML

Informe a seu IdP de SAML que a AWS é a sua provedora de serviços instalando o arquivo `saml-metadata.xml` encontrado em `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` ou`https://signin.aws.amazon.com/static/saml-metadata.xml`. Se a criptografia SAML for necessária, o arquivo poderá ser encontrado em `https://region-code.signin.aws.amazon.com/static/saml/SAMLSP4SHN3UIS2D558H46/saml-metadata.xml`.

Para obter uma lista dos possíveis valores de *region-code* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS).

A forma de instalação desse arquivo depende do IdP. Alguns provedores dão a opção de digitar o URL, e o IdP obtém e instala o arquivo para você. Outros exigem que você baixe o arquivo pelo URL e forneça como arquivo local. Consulte a documentação do IdP para obter detalhes, ou consulte Integrar provedores de soluções SAML de terceiros com a AWS para obter os links para a documentação da web para muitos dos provedores SAML com suporte.

Você também configura as informações que o IdP deverá transmitir como atributos SAML para a AWS, como parte da resposta de autenticação. A maior parte dessas informações aparece na AWS como chaves de contexto de condição que você pode avaliar em suas políticas. Essas chaves de condição garantem que somente usuários autorizados nos contextos corretos recebam permissão para acessar os seus recursos da AWS. Você pode especificar janelas de tempo que restrinjam quando o console pode ser usado. Você também pode especificar o tempo máximo (até 12 horas) que os usuários podem acessar o console antes de precisar atualizar suas credenciais. Para obter mais detalhes, consulte Configurar declarações SAML para a resposta de autenticação.

Visualizar uma resposta do SAML no navegador

Os procedimentos a seguir descrevem como visualizar a resposta do SAML do seu provedor de serviços no navegador ao solucionar um problema relacionado ao SAML 2.0.

Para todos os navegadores, vá para a página onde você possa reproduzir o problema. Em seguida, siga as etapas para o navegador apropriado:

Tópicos

- Google Chrome
- Mozilla Firefox
- Apple Safari
- O que fazer com a resposta do SAML codificada pelo Base64

Google Chrome

Para visualizar uma resposta do SAML no Chrome

Essas etapas foram testadas usando a versão 106.0.5249.103 (compilação oficial) (arm64) do Google Chrome. Caso você use outra versão, talvez seja necessário adaptar as etapas adequadamente.

1. Pressione F12 para iniciar o console Developer Tools (Ferramentas de desenvolvimento).
2. Selecione a guia Network (Rede) e, em seguida, selecione Preserve log (Preservar log) no canto superior esquerdo da janela Developer Tools (Ferramentas de desenvolvimento).
3. Reproduza o problema.
4. (Opcional) Se a coluna Método não estiver visível no painel de log Rede de Ferramentas de desenvolvimento, abra o menu de contexto (clique com o botão direito) em qualquer rótulo de coluna e escolha Método para adicionar a coluna.
5. Procure uma publicação SAML no painel de log Network (Rede) de Developer Tools (Ferramentas de desenvolvimento). Selecione essa linha e visualize a guia Payload (Carga) na parte superior. Procure o elemento SAMLResponse que contém a solicitação codificada. O valor associado é a resposta codificada pelo Base64.

Mozilla Firefox

Para visualizar uma resposta do SAML no Firefox

Esse procedimento foi testado na versão 105.0.3 (64 bits) do Mozilla Firefox. Caso você use outra versão, talvez seja necessário adaptar as etapas adequadamente.

1. Pressione F12 para iniciar o console Ferramentas de desenvolvimento na Web.
2. Selecione a guia Rede.
3. Na parte superior direita da janela Web Developer Tools (Ferramentas de desenvolvimento na Web), escolha opções (o pequeno ícone de engrenagem). Selecione Persist logs (Persistir logs).
4. Reproduza o problema.
5. (Opcional) Se a coluna Método não estiver visível no painel de log Rede de Ferramentas de desenvolvimento na Web, abra o menu de contexto (clique com o botão direito) em qualquer rótulo de coluna e escolha Método para adicionar a coluna.
6. Procure um POST SAML na tabela. Selecione essa linha e, em seguida, visualize a guia Request (Solicitação) e encontre o elemento SAMLResponse. O valor associado é a resposta codificada pelo Base64.

Apple Safari

Para visualizar uma resposta do SAML no Safari

Essas etapas foram testadas usando a versão 16.0 (17614.1.25.9.10, 17614) do Apple Safari. Caso você use outra versão, talvez seja necessário adaptar as etapas adequadamente.

1. Habilite o Inspetor Web no Safari. Abra a janela Preferências, selecione a guia Avançado e Mostrar menu de desenvolvimento na barra de menus.
2. Agora você pode abrir o Inspector Web. Escolha Develop (Desenvolver) na barra de menu e selecione Show Web Inspector (Mostrar o Inspetor da Web).
3. Selecione a guia Rede.
4. No canto superior esquerdo da janela Web Inspector (Inspetor da Web), escolha opções (o pequeno ícone de círculo contendo três linhas horizontais). Selecione Preserve Log (Preservar log).
5. (Opcional) Se a coluna Método não estiver visível no painel de log Rede de Inspetor da Web, abra o menu de contexto (clique com o botão direito) em qualquer rótulo de coluna e escolha Método para adicionar a coluna.

6. Reproduza o problema.
7. Procure um POST SAML na tabela. Selecione essa linha e visualize a guia Headers (Cabeçalhos).
8. Procure o elemento SAMLResponse que contém a solicitação codificada. Desça até encontrar `Request Data` com o nome `SAMLResponse`. O valor associado é a resposta codificada pelo Base64.

O que fazer com a resposta do SAML codificada pelo Base64

Depois que você encontrar o elemento de resposta SAML codificado em Base64 no navegador, copie-o e use a ferramenta de decodificação Base-64 de sua preferência para extrair a resposta marcada com XML.

> **Dica de segurança**
>
> Como os dados da resposta do SAML que você está visualizando podem conter dados de segurança confidenciais, recomendamos que não use um decodificador base64 online. Em vez disso, use uma ferramenta instalada no seu computador local que não envie seus dados do SAML pela rede.

Opção integrada para sistemas do Windows (PowerShell):

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Opção integrada para sistemas MacOS e Linux:

```
$ echo "base64encodedtext" | base64 --decode
```

Revisar os valores no arquivo decodificado

Revise os valores no arquivo de resposta SAML decodificado.

- Verifique se o valor do atributo saml:nameID corresponde ao nome de usuário do usuário autenticado.
- Revise o valor de https://aws.amazon.com/SAML/Attributes/Role. O provedor de ARN e SAML diferencia maiúsculas de minúsculas, e o ARN deve corresponder ao recurso em sua conta.

- Revise o valor de https://aws.amazon.com/SAML/Attributes/RoleSessionName. O valor deverá corresponder ao valor na regra de declaração.
- Se você configurar o valor do atributo para um endereço de e-mail ou nome de conta, verifique se os valores estão corretos. Os valores deverão corresponder ao endereço de e-mail ou nome da conta do usuário autenticado.

Verificar se há erros e confirmar a configuração

Verifique se os valores contêm erros e confirme se as configurações a seguir estão corretas.

- As regras de declaração atendem aos elementos necessários e todos os ARNs estão corretos. Para obter mais informações, consulte Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações.
- Você fez o upload do arquivo de metadados mais recente do seu IdP para o AWS em seu provedor de SAML. Para obter mais informações, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS.
- Você configurou corretamente a política de confiança do perfil do IAM. Para obter mais informações, consulte Métodos para assumir um perfil.

Federação de identidades da AWS para serviços externos

A federação de identidades de saída do IAM permite que suas workloads da AWS acessem com segurança serviços externos sem armazenar credenciais de longo prazo. Suas workloads da AWS podem solicitar JSON Web Tokens (JWTs) de curta duração do AWS Security Token Service (AWS STS) chamando a API GetWebIdentityToken. Esses tokens são assinados criptograficamente, verificáveis publicamente e contêm um conjunto abrangente de dados que comprovam a identidade da sua workload da AWS em serviços externos. Você pode usar esses tokens com uma ampla variedade de provedores de nuvem terceiros, plataformas SaaS e aplicações auto-hospedadas. Os serviços externos verificam a autenticidade do token usando as chaves de verificação da AWS publicadas em endpoints conhecidos e usam as informações nos tokens para tomar decisões de autenticação e autorização.

A federação de identidades de saída elimina a necessidade de armazenar credenciais de longo prazo, como chaves de API ou senhas, no código da aplicação ou nas variáveis de ambiente, melhorando sua postura de segurança. Você pode controlar o acesso à geração de tokens e aplicar propriedades de tokens, como algoritmos de assinatura, públicos permitidos e a duração, usando

políticas do IAM. Todas as solicitações de token são registradas em log na AWS, fornecendo trilhas de auditoria completas para monitoramento de segurança e relatórios de conformidade. Você também pode personalizar tokens com tags que aparecem como reivindicações personalizadas, permitindo que serviços externos implementem um controle de acesso refinado e baseado em atributos.

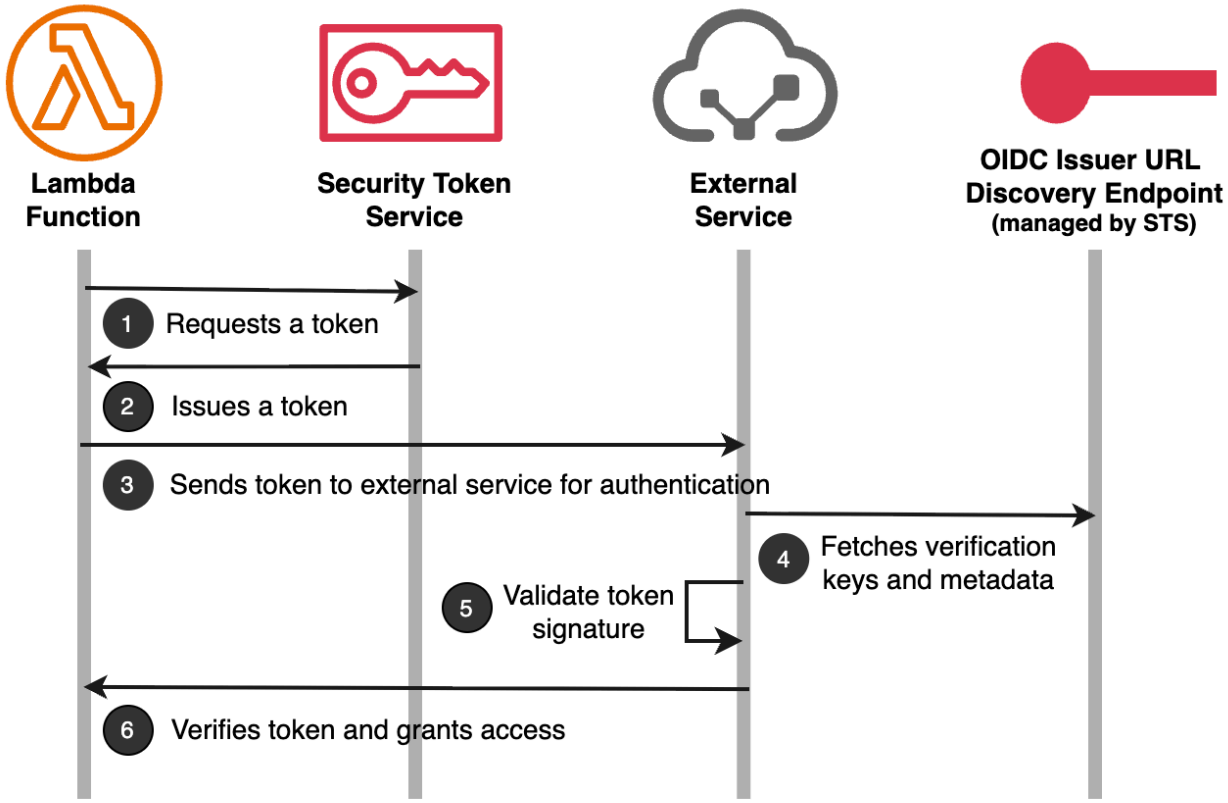
Para obter instruções de configuração da federação de saída, consulte Conceitos básicos da federação de identidades de saída. Para obter mais informações sobre o conteúdo do token, consulte Análise de reivindicações de tokens. Para obter instruções sobre como controlar quais workloads podem solicitar tokens, consulte Controle do acesso com políticas do IAM.

Casos de uso comuns

Usando a federação de identidades de saída, suas workloads da AWS podem:

- Acessar recursos e serviços em provedores de nuvem externos. Por exemplo, uma função do Lambda que processa dados pode gravar resultados no serviço de armazenamento de um provedor de nuvem externo ou consultar seu banco de dados.
- Integrar-se com provedores externos de software como serviço (SaaS) para analytics, processamento de dados, monitoramento etc. Por exemplo, suas funções do Lambda podem enviar métricas para plataformas de observabilidade.
- Autenticar-se com suas próprias aplicações hospedadas na AWS, em provedores de nuvem externos ou em data centers on-premises, permitindo arquiteturas híbridas e multinuvem seguras. Por exemplo, suas workloads da AWS podem interagir com aplicações em contêineres em execução no seu cluster do Kubernetes on-premises.

Como funciona



1. A função do Lambda chama a API GetWebIdentityToken para solicitar um JSON Web Token (JWT) do AWS Security Token Service (AWS STS).
2. AWS STSO valida a solicitação e retorna um JWT assinado para a função do Lambda.
3. A função do Lambda envia o JWT para o serviço externo.
4. O serviço externo extrai o URL do emissor do token, verifica se ele corresponde a um emissor confiável conhecido e obtém os metadados e as chaves de verificação da AWS do endpoint de descoberta do OIDC.
5. O serviço externo usa as chaves de verificação para verificar a assinatura do token e valida as reivindicações como prazo de expiração, assunto e público.
6. Após a validação com êxito, o serviço externo concede acesso à função do Lambda.

Conceitos básicos da federação de identidades de saída

Este guia mostra como habilitar a federação de identidades de saída para sua conta da AWS e obter seu primeiro JSON Web Token (JWT) (usando a API GetWebIdentityToken). Você habilitará o

recurso, estabelecerá uma relação de confiança com um serviço externo, configurará as permissões do IAM e solicitará um token usando a AWS CLI ou o AWS SDK para Python (Boto3).

Pré-requisitos

Antes de começar, você deve ter o seguinte:

- Versão mais recente da AWS CLI ou do Python 3.8 (ou posterior) e do Boto3 instalado (para exemplos do AWS SDK)
- Uma conta de serviço externa em que você pode configurar relações de confiança (como um provedor de nuvem externo, um provedor de SaaS ou uma aplicação de teste)

> **Note**
>
> - A API `GetWebIdentityToken` não está disponível no endpoint global do STS.
> - Os JSON Web Tokens (JWTs) gerados pela API `GetWebIdentityToken` não podem ser usados para a federação do OpenID Connect (OIDC) na AWS (por meio da API AssumeRoleWithWebIdentity).

Habilitar a federação de identidades de saída para sua conta

Você deve habilitar a federação de identidades de saída antes de solicitar tokens. Você pode habilitar o recurso usando o Console de Gerenciamento da AWS, ou programaticamente usando a API EnableOutboundWebIdentityFederation.

Usar a AWS CLI

```
aws iam enable-outbound-web-identity-federation
```

Uso do AWS SDK para Python

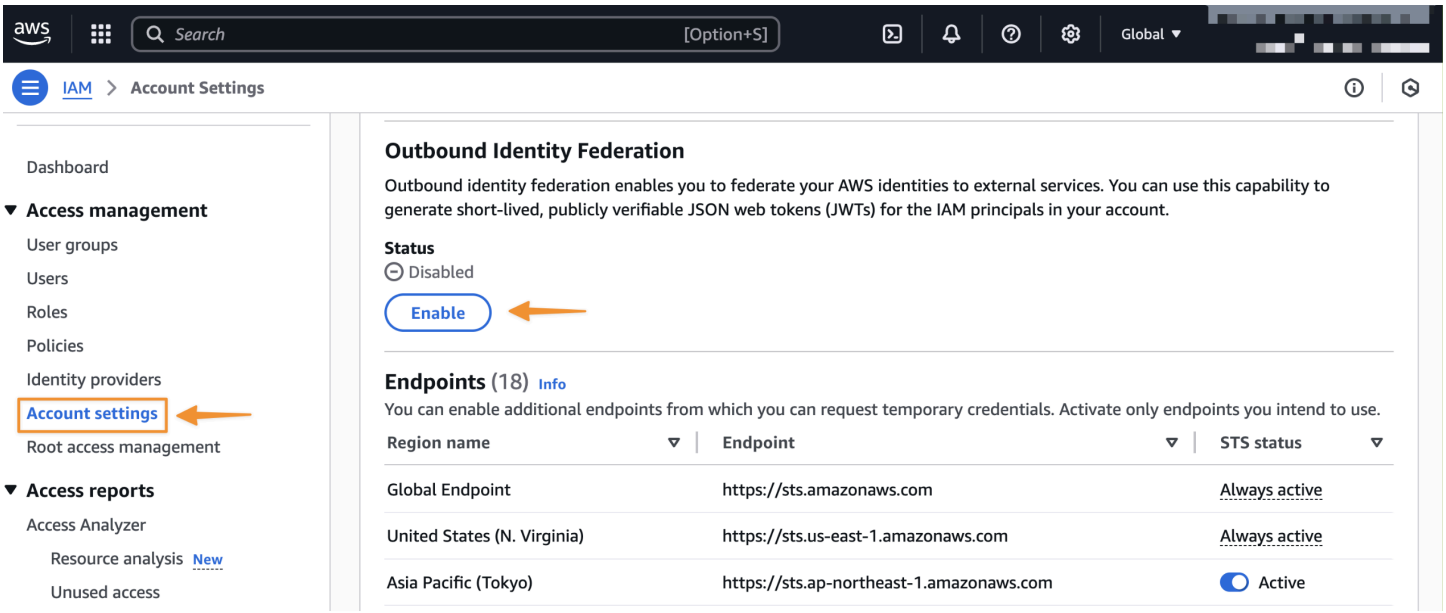
```
import boto3

# Create IAM client
iam_client = boto3.client('iam')
```

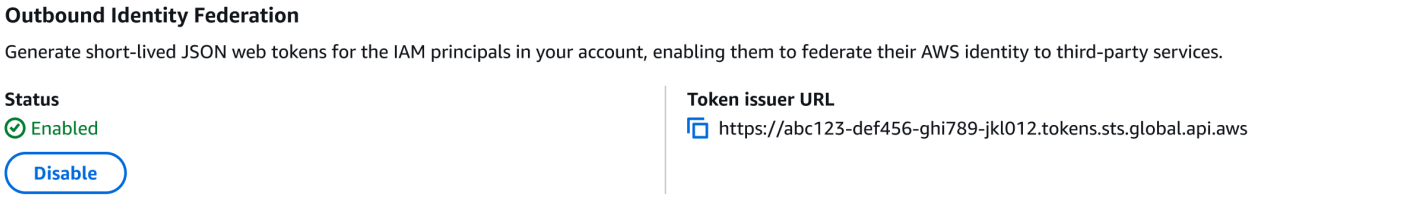
```
# Enable outbound identity federation
response = iam_client.enable_outbound_web_identity_federation()
print(f"Feature enabled. Issuer URL: {response['IssuerUrl']}")
print(f"Status: {response['Status']}")
```

Usar o console AWS

Navegue até IAM e selecione Configurações da conta na seção Gerenciamento de acesso do menu de navegação à esquerda.



Depois de habilitar o recurso, anote o URL do emissor específico da sua conta. Você usará esse URL ao configurar relações de confiança em serviços externos. Você também pode recuperar esse URL do emissor, conforme necessário, usando a API GetOutboundWebIdentityFederationInfo.



Estabelecer uma relação de confiança no serviço externo

Configure o serviço externo para confiar e aceitar os tokens emitidos por sua conta da AWS. As etapas específicas variam de acordo com o serviço, mas geralmente envolvem:

- Registrar o URL do emissor da sua conta da AWS como um provedor de identidades confiável

- Configurar quais reivindicações validar (público, padrões de assunto)
- Mapear reivindicações de tokens para permissões no serviço externo

Consulte a documentação do serviço externo para obter instruções detalhadas de configuração.

Configurar permissões do IAM

Crie uma política do IAM que conceda permissão para chamar a API GetWebIdentityToken e anexe a política a um perfil do IAM que precisa gerar tokens.

Este exemplo de política concede acesso à geração de tokens com restrições específicas. Ela permite solicitar tokens somente para “https://api.example.com” como público e aplica uma vida útil máxima do token de cinco minutos (300 segundos). Consulte the section called “Chaves de condição do IAM” para obter uma lista das chaves de condição que você pode usar para aplicar as propriedades do token.

Exemplo de política do IAM

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "sts:IdentityTokenAudience": "https://api.example.com"
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                }
            }
        }
    ]
}
```

Solicitar seu primeiro JSON Web Token (JWT)

Você pode solicitar um JSON Web Token usando a API GetWebIdentityToken. Você pode especificar os seguintes parâmetros ao chamar a API:

- Audience (obrigatório): o destinatário pretendido do token. Esse valor preenche a reivindicação “aud” no JWT. Os serviços externos validam essa reivindicação para certificar-se de que o token foi destinado a eles.
- SigningAlgorithm (obrigatório): o algoritmo criptográfico usado para assinar o token. Os valores válidos são ES384 e RS256. Use o ES384 para obter segurança e desempenho ideais, ou o RS256 para uma compatibilidade mais ampla com sistemas que não são compatíveis com o ECDSA.
- DurationSeconds (opcional): a vida útil do token em segundos. Os valores válidos variam de 60 a 3600. O padrão é 300 (5 minutos). Recomendamos uma vida útil mais curta do token para aumentar a segurança.
- Tags (opcional): uma lista de pares de chave/valor a serem incluídos como reivindicações personalizadas no token. Os serviços externos podem usar essas reivindicações para uma autorização refinada.

A API retorna os seguintes campos:

- IdentityToken: o JWT assinado como uma string codificada em base64url. Inclua esse token nas reivindicações para serviços externos.
- Expiration: é um carimbo de data e hora UTC para quando o token expirar.

Usar a AWS CLI

```
aws sts get-web-identity-token \
    --audience "https://api.example.com" \
    --signing-algorithm ES384 \
    --duration-seconds 300 \
    --tags Key=team,Value=data-engineering \
           Key=environment,Value=production \
           Key=cost-center,Value=analytics
```

Uso do AWS SDK para Python

```
import boto3

sts_client = boto3.client('sts')

response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
```

```
    DurationSeconds=300,
    SigningAlgorithm='RS256',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'environment', 'Value': 'production'},
        {'Key': 'cost-center', 'Value': 'analytics'}
    ]
)

token = response['WebIdentityToken']
```

Você também pode decodificar o JWT para inspecionar seu conteúdo usando bibliotecas JWT padrão como PyJWT, Python-jose para Python, Nimbus JOSE+JWT para Java ou depuradores como jwt.io. Consulte the section called “Análise de reivindicações de tokens” para obter mais informações sobre as reivindicações incluídas no token.

Usar o token com um serviço externo

Depois de receber o token, inclua-o nas solicitações para o serviço externo. O método varia de acordo com o serviço, mas a maioria dos serviços aceita tokens no cabeçalho de autorização. O serviço externo deve implementar uma lógica de validação de tokens que obtenha as chaves JWKS do endpoint conhecido do emissor, verifique a assinatura do token e valide as reivindicações essenciais antes de conceder acesso às suas workloads da AWS.

Obter chaves de verificação e metadados dos endpoints do OpenID Connect (OIDC)

O URL exclusivo do emissor da sua conta da AWS hospeda os endpoints de descoberta do OpenID Connect (OIDC) que contêm as chaves de verificação e os metadados necessários para a verificação do token.

O URL do endpoint de descoberta do OIDC contém metadados que alguns provedores usam para verificar os tokens. Está disponível em:

```
{issuer_url}/.well-known/openid-configuration
```

O endpoint do JWKS (JSON Web Key Set) contém as chaves usadas para verificar as assinaturas de tokens. Está disponível em:

```
{issuer_url}/.well-known/jwks.json
```

Obter o JWKS usando curl

```
curl https://{issuer_url}/.well-known/jwks.json
```

Resposta:

```
{
  "keys": [
    {
      "kty": "EC",
      "use": "sig",
      "kid": "key-id-1",
      "alg": "ES384",
      "crv": "P-384",
      "x": "base64-encoded-x-coordinate",
      "y": "base64-encoded-y-coordinate"
    },
    {
      "kty": "RSA",
      "use": "sig",
      "kid": "key-id-2",
      "n": "base64-encoded-modulus",
      "e": "AQAB"
    }
  ]
}
```

Uso do AWS SDK para Python

```
import requests

# Fetch Openid Configuration
open_id_config_response = requests.get("https://{issuer_url}/.well-known/openid-
configuration")
open_id_config = open_id_config_response.json()

# Fetch JWKS
jwks_response = requests.get("https://{issuer_url}/.well-known/jwks.json")
jwks = jwks_response.json()
```

Recomendamos armazenar essas chaves em cache para evitar ter de obtê-las em cada verificação de token.

Validações essenciais de reivindicações

- Subject (sub): verifique se a reivindicação do assunto contém o padrão de ARN da entidade principal esperada do IAM.
- Expiration (exp): verifique se o token não expirou. As bibliotecas JWT normalmente lidam com isso automaticamente.
- Audience (aud): verifique se o público corresponde ao valor esperado. Isso evita que tokens destinados a outros serviços sejam usados com os seus.
- Issuer (es): verifique se o emissor corresponde às contas da AWS em que você confia. Mantenha uma lista de URLs de emissores confiáveis.

Sempre que possível, você deve validar reivindicações adicionais específicas da AWS para implementar um controle de acesso refinado em seu serviço externo. Por exemplo, valide a reivindicação org_id para restringir o acesso às entidades principais do IAM em sua organização da AWS, verifique principal_tags para aplicar o controle de acesso baseado em atributos (como permitir somente ambientes de produção ou equipes específicas) ou verifique as reivindicações de contexto de sessão, como lambda_source_function_arn ou ec2_instance_source_vpc, para restringir o acesso com base no recurso computacional. Consulte Análise das reivindicações de tokens para obter uma lista completa das reivindicações incluídas no token.

Análise de reivindicações de tokens

Quando você chama a API GetWebIdentityToken, o AWS Security Token Service retorna um JSON Web Token (JWT) assinado que contém um conjunto de reivindicações que representam a identidade da entidade principal do IAM. Esses tokens são compatíveis com a RFC 7519. Compreender a estrutura e o conteúdo desses tokens ajuda você a implementar fluxos de autenticação seguros, configurar validações de reivindicações apropriadas em serviços externos e usar com eficácia reivindicações personalizadas para um controle de acesso refinado.

O JWT inclui reivindicações padrão do OpenID Connect (OIDC), como subject (“sub”), audience (“aud”), issuer (“iss”), para facilitar a interoperabilidade entre diferentes serviços externos. O AWS STS preenche o token com reivindicações específicas de identidade da AWS (como as tags da entidade principal e do ID da conta da AWS) e reivindicações de contexto de sessão (como ARNs de instância do EC2), quando aplicável. Você também pode adicionar reivindicações personalizadas ao token transmitindo-as como tags de solicitação para a API GetWebIdentityToken. As reivindicações específicas de identidade da AWS, as reivindicações de contexto da sessão e as reivindicações personalizadas estão aninhadas no namespace “https://sts.amazonaws.com/” no token.

Consulte o exemplo de token abaixo para ver uma lista das reivindicações incluídas no token. Observe que todas essas reivindicações podem não estar presentes em um token ao mesmo tempo.

```
{
  "iss": "https://abc123-def456-ghi789-jkl012.tokens.sts.global.api.aws",
  "aud": "https://api.example.com",
  "sub": "arn:aws:iam::123456789012:role/DataProcessingRole",
  "iat": 1700000000,
  "exp": 1700000900,
  "jti": "xyz123-def456-ghi789-jkl012",
  "https://sts.amazonaws.com/": {
    "aws_account": "123456789012",
    "source_region": "us-east-1",
    "org_id": "o-abc1234567",
    "ou_path": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/",
    "principal_tags": {
      "environment": "production",
      "team": "data-engineering",
      "cost-center": "engineering"
    },
    "lambda_source_function_arn": "arn:aws:lambda:us-
east-1:123456789012:function:process-data",
    "request_tags": {
        "job-id": "job-2024-001",
        "priority": "high",
        "data-classification": "sensitive"
    }
  }
}
```

Reivindicações padrão

As reivindicações padrão do OIDC presentes nos tokens facilitam a interoperabilidade com uma ampla gama de serviços externos. Essas reivindicações podem ser validadas usando a maioria das bibliotecas JWT.

Reivindicar	Name (Nome)	Descrição	Valor de exemplo
iss	Emissor	URL do emissor específico da sua conta. Os serviços externos	https://abc123-def456-ghi789-jkl012.

Reivindicar	Name (Nome)	Descrição	Valor de exemplo
		validam essa reivindicação para garantir que ela corresponda ao emissor confiável.	tokens.sts.global.api.aws
aud	Público	O destinatário pretendido para o token especificado na solicitação GetWebIdentityToken.	https://api.example.com
sub	Assunto	O ARN da entidade principal do IAM que solicitou o token.	arn:aws:iam::123456789012:role/DataProcessingRole
iat	Emitido em	Valor NumericDate que identifica a hora em que o JWT foi emitido.	1700000000
exp	Expiração	Valor NumericDate que identifica o prazo de expiração, após o qual o JWT NÃO DEVE ser aceito para processamento.	1700000900
jti	ID do JWT	Identificador exclusivo dessa instância de token.	xyz123-def456-ghi789-jkl012

Reivindicações personalizadas

Além das reivindicações padrão do OIDC, o AWS STS adiciona reivindicações sobre a identidade e o contexto da sessão, quando aplicável. Você também pode adicionar suas próprias reivindicações ao token ao transmiti-las como tags de solicitação. As reivindicações personalizadas estão aninhadas no namespace https://sts.amazonaws.com/.

AWSReivindicações de identidade da

Essas reivindicações fornecem informações detalhadas sobre sua conta da AWS, estrutura organizacional e entidade principal do IAM.

Reivindicar	Descrição	Mapas para a chave de condição	Valor de exemplo
aws_account	O ID da sua conta da AWS.	aws:PrincipalAccount	123456789012
source_region	A região da AWS em que o token foi solicitado	aws:RequestedRegion	us-east-1
org_id	Seu ID do AWS Organizations (se a sua conta fizer parte de uma organização)	aws:PrincipalOrgID	o-abc1234567
ou_path	O caminho da sua unidade organizacional (se aplicável)	aws:PrincipalOrgPaths	o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
principal_tags	Tags anexadas à entidade principal do IAM ou à sessão de perfil assumido. Quando um token é solicitado em que a entidade principal do IAM solicitante tem tags da entidade principal e de sessão, as tags de sessão estarão presentes no JWT.	aws:PrincipalTag/<tag-key>	{"environment": "production", "team": "data-engineering", "cost-center":"engineering"}

Reivindicações de contexto de sessão

Estas reivindicações fornecem informações sobre o ambiente computacional e a sessão em que a solicitação de token foi originada. O AWS STS inclui automaticamente essas reivindicações, quando aplicável, com base no contexto da sessão da entidade principal solicitante.

Reivindicar	Descrição	Mapas para a chave de condição	Valor de exemplo
original_session_exp	Quando as credenciais da sessão de perfil original expirarão (para perfis assumidos)	N/D	2024-01-15T10:00:00Z
federated_provider	O nome do provedor de identidade para sessões federadas	aws:FederatedProvider	arn:aws:iam::111122223333:oidc-provider/your_oidc_provider
identity_store_user_id	ID do usuário do Centro de Identidade do IAM	identitystore:UserId	user-abc123def456
identity_store_arn	ARN do armazenamento de identidades do Centro de Identidade do IAM	identitystore:IdentityStoreArn	arn:aws:identitystore::123456789012:identitystore/d-abc1234567
ec2_source_instance_arn	ARN da instância solicitante do EC2	ec2:SourceInstanceArn	arn:aws:ec2:us-east-1:123456789012:instance/i-abc123def456
ec2_instance_source_vpc	ID da VPC em que as credenciais de perfil do EC2 foram entregues	aws:Ec2InstanceSourceVpc	vpc-abc123def456
ec2_instance_source_private_ipv4	Endereço IPv4 privado da instância do EC2	aws:Ec2InstanceSourcePrivateIPv4	10.0.1.25
ec2_role_delivery	Versão do serviço de metadados da instância	ec2:RoleDelivery	2

Reivindicar	Descrição	Mapas para a chave de condição	Valor de exemplo
source_identity	Identidade de origem definida pela entidade principal	aws:SourceIdentity	admin-user
lambda_source_function_arn	O ARN da função do Lambda chamadora	lambda:SourceFunctionArn	arn:aws:lambda:us-east-1:123456789012:function:my-function
glue_credential_issuing_service	AWSIdentificador do serviço do Glue para trabalhos do Glue	glue:CredentialIssuingService	glue.amazonaws.com

Tags da solicitação

Você pode adicionar reivindicações personalizadas aos tokens especificando tags na solicitação da API GetWebIdentityToken. Essas reivindicações aparecem no campo request_tags no token e permitem que você transmita informações específicas que os serviços externos podem usar para decisões de autorização refinadas. Você pode especificar até 50 tags por solicitação.

Exemplo de solicitação:

```
response = sts_client.get_web_identity_token(
    Audience=['https://api.example.com'],
    SigningAlgorithm='ES384',
    Tags=[
        {'Key': 'team', 'Value': 'data-engineering'},
        {'Key': 'cost-center', 'Value': 'analytics'},
        {'Key': 'environment', 'Value': 'production'}
    ]
)
```

Reivindicações resultantes no token:

```
{
```

```
  "request_tags": {
    "team": "data-engineering",
    "cost-center": "analytics",
    "environment": "production"
  }
}
```

Controle do acesso com políticas do IAM

O IAM fornece vários tipos de políticas para controlar o acesso ao recurso de federação de identidades de saída. Você pode usar políticas baseadas em identidade para controlar quais entidades principais do IAM podem solicitar tokens e aplicar propriedades específicas de tokens, como público, vida útil e algoritmos de assinatura. As políticas de controle de serviços (SCPs) permitem que você aplique restrições em toda a organização à geração de tokens em todas as contas no seu AWS Organizations. As políticas de controle de recursos (RCPs) controlam o acesso no nível do recurso. Você também pode usar políticas de endpoint da VPC para restringir quais entidades principais podem acessar a API `GetWebIdentityToken` do AWS STS por meio de seus endpoints da VPC, adicionando controles em nível de rede à sua postura de segurança. Esta seção explica como implementar controles de acesso refinados usando esses tipos de política e chaves de condição.

Para solicitar tokens de identidade, uma entidade principal do IAM deve ter a permissão `sts:GetWebIdentityToken`. Conceda essa permissão por meio de políticas de identidade anexadas aos usuários ou perfis do IAM. Para permitir que tags (pares de chave/valor) sejam transmitidas para a chamada de GetWebIdentityToken, a entidade principal do IAM deve ter a permissão `sts:TagGetWebIdentityToken`.

- Use a chave de condição sts:IdentityTokenAudience para limitar quais serviços externos podem receber tokens.
- Use a chave de condição sts:DurationSeconds para aplicar a vida útil máxima do token.
- Use a chave de condição sts:SigningAlgorithm para exigir algoritmos criptográficos específicos.
- Use a chave de condição aws:RequestTag para comparar o par de chave/valor da tag que foi transmitido na solicitação com o par de tags especificado na política.
- Use a chave de condição aws:RequestTag para comparar as chaves de tags em uma solicitação com as chaves especificadas na política.

Consulte as chaves de condição do IAM e AWS STS para saber mais sobre as chaves de condição disponíveis nas políticas do IAM.

Esse exemplo de política de identidade combina várias chaves de condição:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowTokenGenerationWithRestrictions",
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api1.example.com",
                        "https://api2.example.com"
                    ]
                },
                "NumericLessThanEquals": {
                    "sts:DurationSeconds": 300
                },
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

Práticas recomendadas

Siga essas recomendações para federar com segurança suas identidades da AWS em serviços externos.

- Use tokens com vida útil curta: solicite tokens com a vida útil mais curta que atenda às suas necessidades operacionais.
- Implemente o acesso com privilégio mínimo e restrinja as propriedades do token com as políticas do IAM: conceda a permissão `sts:GetWebIdentityToken` somente às entidades principais do IAM que a exijam. Use chaves de condição para especificar algoritmos de assinatura, públicos permitidos de tokens e vida útil máxima de tokens, conforme necessário.

- Valide reivindicações em serviços externos: por questões de segurança, sempre valide as reivindicações relevantes, como subject (“sub”), audience (“aud”) etc. para garantir que elas correspondam aos valores esperados. Valide as reivindicações personalizadas sempre que possível para permitir decisões de autorização detalhadas em serviços externos.

Credenciais de segurança temporárias no IAM

É possível usar o AWS Security Token Service (AWS STS) para criar e fornecer aos usuários confiáveis credenciais de segurança temporárias que podem controlar o acesso aos seus recursos da AWS. As credenciais de segurança temporárias funcionam quase de forma idêntica às credenciais de chave de acesso de longo prazo, com as seguintes diferenças:

- As credenciais de segurança temporárias são de curto prazo, como o nome indica. Elas podem ser configuradas para durar de alguns minutos a várias horas. Depois que as credenciais expiram, a AWS não as reconhece mais ou permite qualquer tipo de acesso de solicitações de API feitas com elas.
- As credenciais de segurança temporárias não são armazenadas com o usuário, mas são geradas dinamicamente e fornecidas ao usuário quando solicitadas. Quando (ou até mesmo antes) as credenciais de segurança temporárias expiram, o usuário pode solicitar novas credenciais, desde que o usuário solicitante ainda tenha permissões para fazê-lo.

Como resultado, as credenciais temporárias apresentam as seguintes vantagens em relação às credenciais de longo prazo:

- Você não tem que distribuir ou incorporar credenciais de segurança da AWS de longo prazo com um aplicativo.
- É possível fornecer acesso aos seus recursos da AWS para os usuários sem a necessidade de definir uma identidade da AWS para eles. As credenciais temporárias são a base para perfis e a federação de identidades.
- As credenciais de segurança temporárias têm vida limitada. Portanto, não é necessário atualizá-las ou explicitamente revogá-las quando elas não forem mais necessárias. Quando as credenciais de segurança temporárias expiram, elas não podem ser reutilizadas. É possível especificar por quanto tempo as credenciais são válidas, até um limite máximo.

AWS STS e regiões da AWS

Credenciais de segurança temporárias são geradas pelo AWS STS. Por padrão, o AWS STS é um serviço global com um único endpoint em `https://sts.amazonaws.com`. No entanto, você também pode optar por fazer chamadas de API do AWS STS para endpoints em qualquer outra região com suporte. Isso pode reduzir a latência (atraso do servidor), enviando as solicitações para servidores em uma região que está geograficamente mais perto de você. Não importa de qual região suas credenciais são, elas funcionam globalmente. Para obter mais informações, consulte Gerenciar o AWS STS em uma Região da AWS.

Cenários comuns para credenciais temporárias

As credenciais temporárias são úteis em cenários que envolvem federação de identidades, delegação, acesso entre contas e funções do IAM.

Federação de identidades

Você pode gerenciar suas identidades de usuários em um sistema externo fora da AWS e conceder acesso aos usuários que fazem login a partir desses sistemas para realizar tarefas da AWS e acessar seu recursos da AWS. O IAM é compatível com dois tipos de federação de identidades. Em ambos os casos, as identidades são armazenadas fora da AWS. A distinção é onde o sistema externo reside, em seu data center ou em um terceiro externo na Web. Para comparar os recursos das credenciais de segurança temporárias para federação de identidades, consulte Compare credenciais do AWS STS.

Para obter mais informações sobre provedores de identidade externos, consulte Federação e provedores de identidade na AWS.

- Federação do OpenID Connect (OIDC): você pode permitir que os usuários façam login usando um provedor de identidade de terceiros conhecido, como Login com Amazon, Facebook, Google ou qualquer provedor compatível com OIDC. Para aplicativo móvel ou web, você não precisa criar um código de login personalizado ou gerenciar suas próprias identidades de usuários. O uso da federação OIDC ajuda a manter sua Conta da AWS segura, pois você não precisa distribuir credenciais de segurança de longo prazo, como chaves de acesso de usuários do IAM, com a aplicação. Para obter mais informações, consulte Federação OIDC.

  AWS STSA federação OIDC do oferece suporte a Login with Amazon, Facebook, Google e qualquer provedor de identidades compatível com OpenID Connect (OIDC).

> **Note**
>
> Para aplicações móveis, recomendamos o uso do Amazon Cognito. Você pode usar esse serviço com SDKs da AWS para desenvolvimento de dispositivos móveis para criar identidades exclusivas para os usuários e autenticá-las para acesso seguro aos recursos da AWS. O Amazon Cognito oferece suporte aos mesmos provedores de identidade do AWS STS e também oferece suporte ao acesso não autenticado (de convidado) e permite que você migre os dados do usuário quando um usuário faz login. O Amazon Cognito também fornece operações de API para sincronização de dados de usuário para que eles sejam preservados à medida que passarem de um dispositivo para outro. Para obter mais informações, consulte Autenticação com o Amplify na documentação do Amplify.

- Federação SAML: é possível autenticar usuários na rede da sua organização e, em seguida, fornecer a eles acesso à AWS sem criar novas identidades da AWS para eles nem exigir que façam login com credenciais de login diferentes. Isso é conhecido como a abordagem de logon único para acesso temporário. O AWS STS é compatível com padrões abertos, como o Security Assertion Markup Language (SAML) 2.0, com o qual você pode usar o Microsoft AD FS para utilizar seu Microsoft Active Directory. Você também pode usar o SAML 2.0 para gerenciar sua própria solução para federação de identidades de usuários. Para obter mais informações, consulte Federação SAML 2.0.
  - Agente de federação personalizado: você pode usar o sistema de autenticação da sua organização para conceder acesso aos recursos da AWS. Para obter um cenário de exemplo, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.
  - Federação com SAML 2.0: você pode usar o sistema de autenticação da sua organização e o SAML para conceder acesso aos recursos da AWS. Para obter mais informações e um cenário de exemplo, consulte Federação SAML 2.0.

Funções para acesso entre contas

Muitas organizações mantêm mais de uma Conta da AWS. Com o uso de funções e acesso entre contas, você pode definir identidades de usuários em uma conta e usar essas identidades para acessar recursos da AWS em outras contas que pertencem à sua organização. Isso é conhecido como a abordagem de delegação para acesso temporário. Para obter mais informações sobre a criação de funções entre contas, consulte Criar um perfil para conceder permissões a um usuário do IAM. Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

Funções do Amazon EC2

Se você executa aplicações em instâncias do Amazon EC2 e essas aplicações precisam de acesso a recursos da AWS, você pode fornecer credenciais de segurança temporárias para suas instâncias ao executá-las. Essas credenciais de segurança temporárias estão disponíveis para todos os aplicativos que são executados na instância, portanto você não precisa armazenar qualquer credencial de longo prazo na instância. Para obter mais informações, consulte Use um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2.

Para saber mais sobre as credenciais de perfil do IAM do Amazon EC2, consulte Perfis do IAM para Amazon EC2 no Guia do usuário do Amazon Elastic Compute Cloud.

Outros serviços da AWS

Você pode usar credenciais de segurança temporárias para acessar a maioria dos serviços da AWS. Para obter uma lista dos serviços que aceitam credenciais de segurança temporárias, consulte AWSServiços da que funcionam com o IAM.

Amostra de aplicações que usam credenciais temporárias

É possível usar o AWS Security Token Service (AWS STS) para criar e fornecer aos usuários confiáveis credenciais de segurança temporárias que podem controlar o acesso aos seus recursos da AWS. Para obter mais informações sobre o AWS STS, consulte Credenciais de segurança temporárias no IAM. Para ver como usar AWS STS para gerenciar credenciais de segurança temporárias, você pode fazer download dos seguintes exemplos de aplicativos que implementam exemplos de cenários completos:

- Habilitar a federação na AWS usando o Windows Active Directory, o ADFS e o SAML 2.0. Demonstra como delegar o acesso usando a federação corporativa na AWS usando Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0 e SAML (Security Assertion Markup Language) 2.0.
- Habilitar o acesso do intermediador de identidades personalizado ao console da AWS. Demonstra como criar um proxy de federação personalizado que permite autenticação única (SSO) para que os usuários do Active Directory existentes possam iniciar sessão no Console de gerenciamento da AWS.
- Como usar o Shibboleth para autenticação única no Console de gerenciamento da AWS.. Mostra como usar o Shibboleth e o SAML para fornecer aos usuários logon único (SSO) ao Console de gerenciamento da AWS.

Exemplos de federação OIDC

Os exemplos de apliacações a seguir ilustram como usar a federação OIDC com provedores como Login with Amazon, Amazon Cognito, Facebook ou Google. Você pode trocar a autenticação desses provedores por credenciais de segurança temporárias da AWS para acessar serviços da AWS.

- Tutoriais do Amazon Cognito: recomendamos que você use o Amazon Cognito com os AWS SDKs para desenvolvimento móvel. O Amazon Cognito é a maneira mais simples de gerenciar identidades para aplicativos móveis e fornece recursos adicionais como sincronização e identidade entre dispositivos. Para obter mais informações sobre o Amazon Cognito, consulte Autenticação com o Amplify na documentação do Amplify.

Recursos adicionais para credenciais de segurança temporárias

Os seguintes cenários e aplicativos podem orientá-lo quanto ao uso de credenciais de segurança temporárias:

- Como integrar o AWS STS SourceIdentity ao seu provedor de identidade. Esta postagem mostra como configurar o atributo do AWS STS `SourceIdentity` ao usar Okta, Ping ou OneLogin como IdP.
- Federação OIDC. Esta seção discute como configurar perfis do IAM ao usar federação OIDC e a API `AssumeRoleWithWebIdentity`.
- Acesso seguro à API com a MFA. Este tópico explica como usar funções para exigir a autenticação multifator (MFA) para proteger ações de API confidenciais em sua conta.

Para obter mais informações sobre políticas e permissões na AWS, consulte os seguintes tópicos:

- Gerenciamento de acesso para recursos da AWS
- Lógica da avaliação de política.
- Gerenciar permissões de acesso aos seus recursos do Amazon S3 no Guia do usuário do Amazon Simple Storage Service.
- Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

Compare credenciais do AWS STS

A tabela a seguir compara os recursos das operações da API no AWS STS que retornam credenciais de segurança temporárias. Para saber mais sobre os diferentes métodos que você pode usar para solicitar credenciais de segurança temporárias ao assumir uma função, consulte Métodos para assumir um perfil. Para saber mais sobre as diferentes operações de API do AWS STS que permitem passar tags de sessão, consulte Passar tags de sessão no AWS STS.

> **Note**
>
> Você pode enviar chamadas de API do AWS STS para um endpoint global ou para um dos endpoints regionais. Se escolher um endpoint mais próximo a você, você poderá reduzir a latência e melhorar a performance de suas chamadas de API. Você também pode optar por direcionar suas chamadas para um endpoint regional alternativo se não puder mais se comunicar com o endpoint original. Se estiver usando um dos vários SDKs da AWS, use o método do respectivo SDK para especificar uma região antes de fazer a chamada de API. Se construir manualmente as solicitações de API HTTP, você deverá direcionar a solicitação para o endpoint correto. Para obter mais informações, consulte a seção AWS STS de Regiões e endpoints e Gerenciar o AWS STS em uma Região da AWS.

API do AWS STS	Quem pode chamar	Vida útil da credencial (mín.| máx.| padrão)	Suporte a MFA[1]	Suporte à política de sessão[2]	Restrições nas credenciais temporárias resultantes
AssumeRole	Usuário do IAM ou função do IAM com credenciais de segurança temporárias existentes	15 min | Configuração de duração máxima da sessão[3] | 1 h	Sim	Sim	Não pode chamar `GetFederationToken` ou `GetSessionToken` .

API do AWS STS	Quem pode chamar	Vida útil da credencial (mín.| máx.| padrão)	Suporte a MFA[1]	Suporte à política de sessão[2]	Restrições nas credenciais temporárias resultantes
AssumeRoleWithSAML	Todo chamador de usuário; deve passar uma resposta de autenticação de SAML que indica a autenticação de um provedor de identidade conhecido	15 min | Configuração de duração máxima da sessão[3] | 1 h	Não	Sim	Não pode chamar `GetFederationToken` ou `GetSessionToken`.
AssumeRoleWithWebIdentity	Qualquer usuário; o chamador deve passar um token JWT compatível com OIDC que indique a autenticação por um provedor de identidade conhecido	15 min | Configuração de duração máxima da sessão[3] | 1 h	Não	Sim	Não pode chamar `GetFederationToken` ou `GetSessionToken`.

API do AWS STS	Quem pode chamar	Vida útil da credencial (mín.| máx.| padrão)	Suporte a MFA[1]	Suporte à política de sessão[2]	Restrições nas credenciais temporárias resultantes
GetFederationToken	Usuário do IAM ou Usuário raiz da conta da AWS	Usuário do IAM: 15 min | 36 h | 12 h Usuário raiz: 15 min | 1 h | 1 h	Não	Sim	Não é possível chamar operações do IAM usando a AWS CLI ou a API da AWS. Essa limitação não se aplica a sessões do console. Não é possível chamar operações do AWS STS, exceto `GetCallerIdentity` .[4] Logon único para console é permitido.[5]
GetSessionToken	Usuário do IAM ou Usuário raiz da conta da AWS	Usuário do IAM: 15 min | 36 h | 12 h Usuário raiz: 15 min | 1 h | 1 h	Sim	Não	Não é possível chamar as operações da API do IAM, a menos que as informações da MFA sejam incluídas na solicitação. Não pode chamar as operações de API do AWS STS exceto `AssumeRole` ou `GetCallerIdentity` . Logon único para console não é permitido.[6]

[1] Compatibildade com MFA. Você pode incluir informações sobre um dispositivo de autenticação multifator (MFA) quando chamar as operações de API AssumeRole e GetSessionToken. Isso

garante que as credenciais de segurança temporárias que resultam da chamada de API possam ser usadas somente pelos usuários que são autenticados com um dispositivo MFA. Para obter mais informações, consulte Acesso seguro à API com a MFA.

[2] Suporte à política de sessão. As políticas de sessão são políticas que você transmite como um parâmetro quando você cria de forma programática uma sessão temporária para uma sessão de usuário federado do AWS STS ou perfil. Esta política limita as permissões da política baseada em identidade da função ou do usuário que são atribuídas à sessão. As permissões da sessão resultam da interseção das políticas baseadas em identidade da entidade e das políticas de sessão. As políticas de sessão não podem ser usadas para conceder mais permissões do que as permitidas pela política baseada em identidade da função que está sendo assumida. Para obter mais informações sobre as permissões de sessão da função, consulte Políticas de sessão.

[3] Configuração de duração máxima da sessão. Use o parâmetro `DurationSeconds` para especificar a duração da sessão da função de 900 segundos (15 minutos) até o valor configurado de duração máxima da sessão para a função. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil.

[4] GetCallerIdentity. Nenhuma permissão é necessária para executar essa operação. Se um administrador adicionar uma política ao seu usuário ou função do IAM que negue explicitamente o acesso à ação `sts:GetCallerIdentity`, você ainda poderá executar esta operação. As permissões não são necessárias porque as mesmas informações são retornadas quando um usuário ou uma função do IAM tem acesso negado. Para visualizar uma resposta de exemplo, consulte Não estou autorizado a executar: iam:DeleteVirtualMFADevice.

[5] Logon único (SSO) para o console. Para dar suporte a SSO, a AWS permite chamar um endpoint da federação (`https://signin.aws.amazon.com/federation`) e passar credenciais de segurança temporárias. O endpoint retorna um token que pode ser usado para construir um URL que assina um usuário diretamente no console sem a necessidade de uma senha. Para obter mais informações, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS e Como habilitar o acesso entre contas ao Console de Gerenciamento da AWS no Blog de segurança da AWS.

[6] Depois de recuperar as credenciais temporárias, você não poderá acessar o Console de gerenciamento da AWS transmitindo as credenciais para o endpoint de logon único de federação. Para obter mais informações, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.

Tokens de portador de serviço

Alguns serviços da AWS exigem que você tenha permissão para obter um token de portador do serviço AWS STS para poder acessar seus recursos de forma programática. Esses serviços são compatíveis com um protocolo que requer que você use um token de portador em vez de usar uma tradicional AWS Signature Version 4 para solicitações de API. Quando você executa operações da AWS CLI ou da API da AWS que exigem tokens de portador, o serviço da AWS solicita um token de portador em seu nome. O serviço fornece o token que você pode usar para executar operações subsequentes nesse serviço.

AWS STSOs tokens de portador do serviço incluem informações de sua autenticação principal original que podem afetar suas permissões. Essas informações podem incluir tags de principal, tags de sessão e políticas de sessão. O ID de chave de acesso do token começa com o prefixo `ABIA`. Isso ajuda você a identificar operações que foram realizadas usando tokens de portador de serviço em seus logs do CloudTrail.

> **Important**
>
> O token de portador pode ser usado apenas para chamadas ao serviço que o gera e na região onde ele foi gerado. Você não pode usar o token de portador para executar operações em outros serviços ou regiões.

Um exemplo de serviço compatível com tokens de portador é o AWS CodeArtifact. Para poder interagir com o AWS CodeArtifact usando um gerenciador de pacotes, como NPM, Maven ou PIP, você deve chamar a operação `aws codeartifact get-authorization-token`. Essa operação retorna um token de portador que você pode usar para executar operações do AWS CodeArtifact. Como alternativa, você pode usar o comando `aws codeartifact login` que conclui a mesma operação e configura seu cliente automaticamente.

Para executar uma ação em um serviço da AWS que gera um token de portador para você, você deverá ter as seguintes permissões em sua política do IAM:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Para obter um exemplo de token de portador do serviço, consulte Usar políticas baseadas em identidade para o AWS CodeArtifact no Guia do usuário do AWS CodeArtifact.

Solicitar credenciais de segurança temporárias

Para solicitar credenciais de segurança temporárias, você pode usar as operações do AWS Security Token Service (AWS STS) na API da AWS. É possível usar operações para criar e fornecer aos usuários confiáveis credenciais de segurança temporárias que podem controlar o acesso aos seus recursos da AWS. Para obter mais informações sobre o AWS STS, consulte Credenciais de segurança temporárias no IAM. Para saber mais sobre os diferentes métodos que você pode usar para solicitar credenciais de segurança temporárias ao assumir uma função, consulte Métodos para assumir um perfil.

Para chamar as operações de API, você pode usar um dos AWS SDKs. Os SDKs estão disponíveis para uma grande variedade de ambientes e linguagens de programação, incluindo Java, .NET, Python, Ruby, Android e iOS. Os SDKs processam tarefas como a assinatura criptográfica de suas solicitações, a solicitação de novas tentativas (se necessário) e o tratamento das respostas de erro. Você também pode usar a API de consulta do AWS STS, descrita na Referência de API do AWS Security Token Service. Por fim, duas ferramentas de linha de comando são suporte aos comandos do AWS STS: a AWS Command Line Interface e o AWS Tools for Windows PowerShell.

As operações da API do AWS STS criam uma nova sessão com credenciais de segurança temporárias que incluem um par de chaves de acesso e um token de sessão. O par de chaves de acesso consiste em um ID de chave de acesso e uma chave secreta. Os usuários (ou um aplicativo que o usuário executa) pode usar essas credenciais para acessar seus recursos.

Você pode criar uma sessão de função e aprovar políticas e tags de sessão programaticamente usando operações de API do AWS STS. As permissões de sessão resultante são a interseção das políticas baseadas em identidade da função e das políticas de sessão. Para obter mais informações sobre políticas de sessão, consulte Políticas de sessão. Para obter mais informações sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

> **Note**
>
> O tamanho do token de sessão que as operações de API do AWS STS retornam não é fixo. É altamente recomendável que você não faça suposições sobre o tamanho máximo. O tamanho típico do token é menos de 4096 bytes, mas pode variar.

Uso do AWS STS com regiões da AWS

Você pode enviar chamadas de API do AWS STS para um endpoint global ou para um dos endpoints regionais. Se escolher um endpoint mais próximo a você, você poderá reduzir a latência e melhorar a performance de suas chamadas de API. Você também pode optar por direcionar suas chamadas para um endpoint regional alternativo se não puder mais se comunicar com o endpoint original.

Se estiver usando um dos vários SDKs da AWS, use o método do respectivo SDK para especificar uma região antes de fazer a chamada de API. Se construir manualmente as solicitações de API HTTP, você deverá direcionar a solicitação para o endpoint correto. Para obter mais informações, consulte a seção AWS STS de Regiões e endpoints e Gerenciar o AWS STS em uma Região da AWS.

A seguir veja as operações de API que podem ser usadas para adquirir credenciais temporárias para o uso em seu ambiente e aplicativos da AWS.

Solicitar credenciais para delegação e federação entre contas por meio de um intermediador de identidades personalizado

A operação da API `AssumeRole` é útil para permitir que os usuários existentes do IAM acessem os recursos da AWS aos quais eles ainda não têm acesso. Por exemplo, o usuário pode precisar de acesso a recursos em outra Conta da AWS. Ela também é útil como um meio de obter acesso privilegiado temporariamente, por exemplo, para fornecer uma autenticação multifator (MFA). Você deve chamar essa API usando credenciais ativas. Para saber quem pode chamar essa operação, consulte Compare credenciais do AWS STS. Para obter mais informações, consulte Criar um perfil para conceder permissões a um usuário do IAM e Acesso seguro à API com a MFA.

Para solicitar credenciais de segurança temporárias para delegação e federação entre contas por meio de um intermediador de identidades personalizado

1. Autentique-se com suas credenciais de segurança da AWS. Essa chamada deve ser feita usando credenciais de segurança da AWS válidas.

2. Chame a operação `AssumeRole`.

O exemplo a seguir mostra uma solicitação e resposta de exemplo usando `AssumeRole`. Este exemplo de solicitação assume a função `demo` para a duração especificada com a política de sessão incluída, etiquetas de sessão, ID externo e identidade-fonte. A sessão resultante é nomeada `John-session`.

Example Exemplo de solicitação

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws:iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17       %22%2C%22Statement%22%3A%5B%7B%22Sid
%22%3A%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*
%22%2C%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

O valor da política mostrado no exemplo anterior é a versão codificada por URL da seguinte política:

JSON

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

O parâmetro `AUTHPARAMS` no exemplo é um espaço reservado para a sua assinatura. Uma assinatura é a informação de autenticação que você deve incluir com as solicitações de API HTTP da AWS. Recomendamos usar os SDKs da AWS para criar solicitações de API. Um dos benefícios de se fazer isso é que os SDKs tratam da assinatura das solicitações por você. Se você tiver que criar e assinar as solicitações de API manualmente, acesse Assinaturas e solicitações da AWS usando

o Signature versão 4 no Referência geral da Amazon Web Services para saber como assinar uma solicitação.

Além das credenciais de segurança temporárias, a resposta inclui o nome de recurso da Amazon (ARN) para o usuário federado e o tempo de expiração das credenciais.

Example Exemplo de resposta

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>34</PackedPolicySize>
<SessionTokenSize>1428</SessionTokenSize>
<SessionTokenUtilization>34</SessionTokenUtilization>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> As políticas de sessão e as tags de sessão aumentam o tamanho do token da sessão. Se o token exceder o limite de tamanho, sua solicitação falhará, mesmo que seu texto simples

esteja dentro dos outros limites. O elemento de resposta `SessionTokenUtilization` mostra qual porcentagem do limite de tamanho o token usa. O elemento de resposta `PackedPolicySize` está obsoleto. Use `SessionTokenUtilization` em vez disso.

Solicitar credenciais por meio de um provedor OIDC

A operação da API `AssumeRoleWithWebIdentity` retorna um conjunto de credenciais de segurança temporárias da AWS em troca de um JSON Web Token (JWT). Isso inclui provedores de identidade públicos, como Login with Amazon, Facebook, Google e provedores que emitem JWTs compatíveis com a descoberta do OpenID Connect (OIDC), como GitHub Actions ou Azure DevOps. Para obter mais informações, consulte Federação OIDC.

> **Note**
>
> As solicitações `AssumeRoleWithWebIdentity` não são assinadas com as credenciais da AWS nem as exigem.

Solicitar credenciais por meio de um provedor OIDC

1. Chame a operação `AssumeRoleWithWebIdentity`.

   Quando você chama `AssumeRoleWithWebIdentity`, a AWS valida o token apresentado verificando a assinatura digital usando chaves públicas disponibilizadas por meio do conjunto de chaves da Web JSON (JWKS) do seu IdP. Se o token for válido e todas as condições estabelecidas na política de confiança do perfil do IAM forem atendidas, a AWS retornará as seguintes informações para você:

   - Um conjunto de credenciais de segurança temporárias. Elas consistem em um ID de chave de acesso, chave de acesso secreta e um token de sessão.
   - O ID da função e o ARN da função assumida.
   - Um valor `SubjectFromWebIdentityToken` que contém o ID de usuário exclusivo.
2. A aplicação pode então usar as credenciais de segurança temporárias que foram retornadas na resposta para fazer chamadas de API da AWS. Esse é o mesmo processo de fazer uma chamada de API da AWS com credenciais de segurança de longo prazo. A diferença é que você deve incluir o token de sessão, o que permite que a AWS verifique se as credenciais de segurança temporárias são válidas.

A aplicação deve armazenar em cache as credenciais retornadas pelo AWS STS e atualizá-las conforme necessário. Se a aplicação foi criada usando um AWS SDK, ele tem provedores de credenciais que podem lidar com as chamadas `AssumeRoleWithWebIdentity` e a atualização de credenciais da AWS antes que elas expirem. Para obter mais informações, consulte AWS SDKs and Tools standardized credential providers no Guia de referência de ferramentas e SDKs da AWS.

Solicitar credenciais por meio de um provedor de identidades SAML 2.0

A operação de API `AssumeRoleWithSAML` retorna um conjunto de credenciais de segurança temporárias para entidades principais federadas SAML que são autenticadas pelo sistema de identidade existente da sua organização. Os usuários também devem usar SAML 2.0 (Security Assertion Markup Language) para passar informações de autenticação e autorização para a AWS. Esta operação de API é útil em organizações que integraram seus sistemas de identidade (como o Windows Active Directory ou OpenLDAP) a um software que pode produzir declarações em SAML. Essa integração fornece informações sobre a identidade e as permissões do usuário (como o Active Directory Federation Services ou o Shibboleth). Para obter mais informações, consulte Federação SAML 2.0.

1. Chame a operação `AssumeRoleWithSAML`.

   Essa é uma chamada não assinada, o que significa que você não precisa autenticar as credenciais de segurança da AWS antes de fazer a solicitação.

   > **Note**
   >
   > Uma chamada para `AssumeRoleWithSAML` não é assinada (criptografada). Portanto, você deve incluir somente as políticas de sessão opcionais se a solicitação é transmitida por meio de um intermediário de confiança. Nesse caso, alguém poderia alterar a política para remover as restrições.

2. Quando você chama `AssumeRoleWithSAML`, a AWS verifica a autenticidade da declaração do SAML. Supondo que o provedor de identidade valide a declaração, a AWS retorna as seguintes informações:

   - Um conjunto de credenciais de segurança temporárias. Elas consistem em um ID de chave de acesso, chave de acesso secreta e um token de sessão.
   - O ID da função e o ARN da função assumida.

- Um valor `Audience` que contém o valor do atributo `Recipient` do elemento `SubjectConfirmationData` da declaração do SAML.
- Um valor `Issuer` que contém o valor do o elemento `Issuer` da declaração do SAML.
- Um elemento `NameQualifier` que contém um valor de hash criado a partir do valor `Issuer`, o ID da Conta da AWS e o nome amigável do provedor de SAML. Quando combinado com o elemento `Subject`, eles podem identificar exclusivamente a entidade principal federada SAML.
- Um elemento `Subject` que contém o valor do elemento `NameID` no elemento `Subject` da declaração do SAML.
- Um elemento `SubjectType` que indica o formato do elemento `Subject`. O valor pode ser `persistent`, `transient`ou o URI `Format` completo dos elementos `Subject` e `NameID` usados em sua declaração do SAML. Para obter mais informações sobre o atributo `NameID` do elemento `Format`, consulte Configurar declarações SAML para a resposta de autenticação.

3. Use as credenciais de segurança temporárias devolvidas na resposta para fazer chamadas de API da AWS. Esse é o mesmo processo de fazer uma chamada de API da AWS com credenciais de segurança de longo prazo. A diferença é que você deve incluir o token de sessão, o que permite que a AWS verifique se as credenciais de segurança temporárias são válidas.

Seu aplicativo deve armazenar as credenciais em cache. Por padrão, as credenciais expiram após uma hora. Se você não estiver usando a ação AmazonSTSCredentialsProvider no SDK da AWS, caberá a você e ao seu aplicativo chamar `AssumeRoleWithSAML` novamente. Chame esta operação para obter um novo conjunto de credenciais de segurança temporárias antes que as antigas expirem.

Solicitar credenciais por meio de um intermediador de identidades personalizado

A operação de API `GetFederationToken` retorna um conjunto de credenciais de segurança temporárias para usuários federados do AWS STS. A API é diferente de `AssumeRole`, em que o período de expiração padrão é significativamente maior (até 12 horas, em vez de 1 hora). Além disso, você pode usar o parâmetro `DurationSeconds` para especificar uma duração para que as credenciais de segurança temporárias permaneçam válidas. As credenciais resultantes são válidas pela duração especificada, de 900 segundos (15 minutos) até 129.600 segundos (36 horas). O período de expiração maior pode ajudar a reduzir o número de chamadas para a AWS porque você não precisa obter novas credenciais com a mesma frequência.

1. Faça a autenticação com as credenciais de segurança da AWS do seu usuário do IAM específico. Essa chamada deve ser feita usando credenciais de segurança da AWS válidas.
2. Chame a operação `GetFederationToken`.

A chamada `GetFederationToken` retorna credenciais de segurança temporárias que consistem em um token de segurança, chave de acesso, chave secreta e expiração. Você pode usar `GetFederationToken` se deseja gerenciar permissões dentro de sua organização (por exemplo, usando o aplicativo de proxy para atribuir permissões).

O exemplo a seguir mostra uma solicitação e resposta de exemplo que usa `GetFederationToken`. Esta solicitação de exemplo agrupa o usuário de chamada pela duração especificada ao ARN da política de sessão e às tags de sessão. A sessão resultante é nomeada `Jane-session`.

Example Exemplo de solicitação

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

O ARN da política mostrado no exemplo anterior inclui o seguinte ARN codificado em URL:

```
arn:aws:iam::123456789012:policy/Role1policy
```

Além disso, observe que o parâmetro `&AUTHPARAMS` no exemplo destina-se a ser o espaço reservado para as informações de autenticação. Esta é a assinatura, que é necessário incluir às solicitações de API HTTP da AWS. Recomendamos usar os SDKs da AWS para criar solicitações de API. Um dos benefícios de se fazer isso é que os SDKs tratam da assinatura das solicitações por você. Se você tiver que criar e assinar as solicitações de API manualmente, acesse Assinaturas e solicitações da AWS usando o Signature versão 4 no Referência geral da Amazon Web Services para saber como assinar uma solicitação.

Além das credenciais de segurança temporárias, a resposta inclui o nome de recurso da Amazon (ARN) para o usuário federado e o tempo de expiração das credenciais.

Example Exemplo de resposta

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>26</PackedPolicySize>
<SessionTokenSize>1105</SessionTokenSize>
<SessionTokenUtilization>26</SessionTokenUtilization>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> As políticas de sessão e as tags de sessão aumentam o tamanho do token da sessão. Se o token exceder o limite de tamanho, sua solicitação falhará, mesmo que seu texto simples esteja dentro dos outros limites. O elemento de resposta `SessionTokenUtilization` mostra qual porcentagem do limite de tamanho o token usa. O elemento de resposta `PackedPolicySize` está obsoleto. Use `SessionTokenUtilization` em vez disso.

A AWS recomenda que você conceda permissões no nível do recurso (por exemplo, você anexa uma política baseada em recurso a um bucket do Amazon S3). Você pode omitir o parâmetro `Policy`. No entanto, se você não incluir uma política para a entidade principal do usuário federado do AWS STS, as credenciais de segurança temporárias não concederão permissões. Neste caso, você deve usar as políticas de recurso para conceder ao usuário federado o acesso aos seus recursos da AWS.

Por exemplo, suponha que o número da sua Conta da AWS seja 111122223333 e você tenha um bucket do Amazon S3 que deseja permitir que Susan acesse. As credenciais de segurança temporárias da Susan não incluem uma política para o bucket. Nesse caso, é necessário certificar-se de que o bucket tem uma política com um ARN que corresponde ao de Susan, como `arn:aws:sts::111122223333:federated-user/Susan`.

Solicitar credenciais para usuários em ambientes não confiáveis

A operação da API `GetSessionToken` retorna um conjunto de credenciais de segurança temporárias para um usuário do IAM existente. Ela é útil para fornecer segurança aprimorada, como, por exemplo, permitir solicitações da AWS somente quando a MFA estiver habilitada para o usuário do IAM. Como as credenciais são temporárias, elas fornecem segurança aprimorada quando você tem um usuário do IAM que acessa seus recursos por meio de um ambiente menos seguro. Exemplos de ambientes menos seguros incluem um dispositivo móvel ou navegador da web.

1. Faça a autenticação com as credenciais de segurança da AWS do seu usuário do IAM específico. Essa chamada deve ser feita usando credenciais de segurança da AWS válidas.
2. Chame a operação `GetSessionToken`.
3. `GetSessionToken` retorna as credenciais de segurança temporárias que consistem em um token de sessão, um ID de chave de acesso e uma chave de acesso secreta.

Por padrão, as credenciais de segurança temporárias de um usuário do IAM são válidas por no máximo 12 horas. Mas você pode solicitar uma duração de, no mínimo, 15 minutos ou, no máximo, 36 horas usando o parâmetro `DurationSeconds`. Por motivos de segurança, um token para um Usuário raiz da conta da AWS é restrito a uma hora de duração.

O exemplo a seguir mostra uma solicitação e resposta de exemplo usando `GetSessionToken`. A resposta também inclui o tempo de expiração das credenciais de segurança temporárias.

Example Exemplo de solicitação

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

O parâmetro AUTHPARAMS no exemplo é um espaço reservado para a sua assinatura. Uma assinatura é a informação de autenticação que você deve incluir com as solicitações de API HTTP da AWS. Recomendamos usar os SDKs da AWS para criar solicitações de API. Um dos benefícios de se fazer isso é que os SDKs tratam da assinatura das solicitações por você. Se você tiver que criar e assinar as solicitações de API manualmente, acesse Assinaturas e solicitações da AWS usando o Signature versão 4 no Referência geral da Amazon Web Services para saber como assinar uma solicitação.

Example Exemplo de resposta

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<SessionTokenSize>856</SessionTokenSize>
<SessionTokenUtilization>20</SessionTokenUtilization>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

Opcionalmente, a solicitação `GetSessionToken` pode incluir os valores `SerialNumber` e `TokenCode` para a verificação da autenticação multifator (MFA) da AWS. Se os valores fornecidos forem válidos, o AWS STS fornecerá credenciais de segurança temporárias que incluem o estado da autenticação MFA. As credenciais de segurança temporárias podem ser usadas para acessar as operações de API protegidas por MFA ou os sites da AWS pelo tempo em que a autenticação MFA for válida.

O exemplo a seguir mostra uma solicitação `GetSessionToken` que inclui um código de verificação de MFA e número de série do dispositivo.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> A chamada para o AWS STS pode ser para o endpoint global ou para qualquer um dos endpoints regionais para os quais você ativar sua Conta da AWS. Para obter mais informações, consulte a seção do AWS STS de Regiões e endpoints.
> O parâmetro `AUTHPARAMS` no exemplo é um espaço reservado para a sua assinatura. Uma assinatura é a informação de autenticação que você deve incluir com as solicitações de API HTTP da AWS. Recomendamos usar os SDKs da AWS para criar solicitações de API. Um dos benefícios de se fazer isso é que os SDKs tratam da assinatura das solicitações por você. Se você tiver que criar e assinar as solicitações de API manualmente, acesse Assinaturas e solicitações da AWS usando o Signature versão 4 no Referência geral da Amazon Web Services para saber como assinar uma solicitação.

Usar credenciais temporárias com recursos da AWS

Você pode usar credenciais de segurança temporárias para fazer solicitações programáticas de recursos da AWS usando a AWS CLI ou API da AWS (usando os SDKs da AWS). As credenciais temporárias fornecem as mesmas permissões que as credenciais de segurança de longo prazo, como as credenciais de usuário do IAM. No entanto, há algumas diferenças:

- Ao fazer uma chamada usando credenciais de segurança temporárias, ela deve incluir um token de sessão, que é retornado junto com essas credenciais temporárias. O AWS usa o token de sessão para validar as credenciais de segurança temporárias.
- As credenciais temporárias expiram após um intervalo especificado. Após a expiração das credenciais temporárias, ocorrerá falha em todas as chamadas que você fizer com essas credenciais, portanto, você deve gerar um novo conjunto de credenciais temporárias. As credenciais temporárias não podem ser estendidas nem renovadas além do intervalo original especificado.
- Ao usar credenciais temporárias para fazer uma solicitação, seu principal pode incluir um conjunto de tags. Essas tags vêm de tags de sessão e tags anexadas à função que você assume. Para obter mais informações sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

Se você usar os SDKs da AWS, a AWS Command Line Interface (AWS CLI) ou o Tools for Windows PowerShell, a maneira de obter e usar credenciais de segurança temporárias será diferente conforme o contexto. Se você estiver executando código, comandos da AWS CLI ou do Tools for Windows PowerShell em uma instância do EC2, poderá aproveitar as funções do Amazon EC2. Caso contrário, é possível chamar uma API do AWS STS para obter as credenciais temporárias e, em seguida, usá-las explicitamente para fazer chamadas de serviços da AWS.

> **Note**
>
> É possível usar o AWS Security Token Service (AWS STS) para criar e fornecer aos usuários confiáveis credenciais de segurança temporárias que podem controlar o acesso aos seus recursos da AWS. Para obter mais informações sobre o AWS STS, consulte Credenciais de segurança temporárias no IAM. AWS STS é um serviço global que tem um endpoint padrão em `https://sts.amazonaws.com`. Esse endpoint fica na região Leste dos EUA (N. da Virgínia)), embora as credenciais obtidas desse e de outros endpoints sejam válidas globalmente. Essas credenciais funcionam com serviços e recursos em qualquer região. Você também pode optar por fazer chamadas de API do AWS STS para endpoints em qualquer uma das regiões com suporte. Isso pode reduzir a latência fazendo as solicitações de servidores em uma região que está geograficamente mais perto de você. Não importa de qual região suas credenciais são, elas funcionam globalmente. Para obter mais informações, consulte Gerenciar o AWS STS em uma Região da AWS.

Sumário

- Uso de credenciais temporárias em instâncias do Amazon EC2
- Uso de credenciais de segurança temporárias com os AWS SDKs
- Uso de credenciais de segurança temporárias com a AWS CLI
- Uso de credenciais de segurança temporárias com operações de API
- Mais informações

Uso de credenciais temporárias em instâncias do Amazon EC2

Se você deseja executar comandos da AWS CLI ou código dentro de uma instância do EC2, a forma recomendada de obter credenciais é usar funções para o Amazon EC2. Você cria uma função do IAM que especifica as permissões a serem concedidas a aplicações em execução nas instâncias do EC2. Ao executar a instância, você associa a função à instância.

Aplicações, a AWS CLI e comandos do Tools for Windows PowerShell executados na instância podem, então, obter credenciais de segurança temporárias automáticas dos metadados da instância. Você não precisa obter explicitamente as credenciais de segurança temporárias. Os AWS SDKs, a AWS CLI e as Tools for Windows PowerShell obtêm automaticamente as credenciais do serviço de metadados da instância (IMDS) do EC2 e as utilizam. As credenciais temporárias têm as permissões que você define para a função que está associada à instância.

Para obter mais informações e exemplos, veja a seguir:

- Usar funções do IAM para conceder acesso a recursos da AWS no Amazon Elastic Compute Cloud — AWS SDK para Java
- Conceder acesso utilizando uma função do IAM — AWS SDK para .NET
- Criar uma função — AWS SDK para Ruby

Uso de credenciais de segurança temporárias com os AWS SDKs

Para usar credenciais de segurança temporárias em código, você chama programaticamente uma API do AWS STS como `AssumeRole` e extrai as credenciais resultantes e o token de sessão. Depois, use esses valores como credenciais em chamadas subsequentes para a AWS. O exemplo a seguir mostra o pseudocódigo de como usar credenciais de segurança temporárias se você estiver usando um SDK da AWS:

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
```

```
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Para obter um exemplo escrito em Python (usando o AWS SDK para Python (Boto)), consulte Alternar para um perfil do IAM (API da AWS). Este exemplo mostra como chamar `AssumeRole` para obter credenciais de segurança temporárias e usar essas credenciais para fazer uma chamada para o Amazon S3.

Para obter detalhes sobre como chamar `AssumeRole`, `GetFederationToken` e outras operações de API, consulte a Referência da API do AWS Security Token Service. Para obter informações sobre como obter as credenciais de segurança temporárias e o token de sessão provenientes do resultado, consulte a documentação do SDK com o qual você está trabalhando. Você pode encontrar a documentação de todos os AWS SDKs na principal página da documentação da AWS, na seção SDKs e toolkits.

É necessário obter um novo conjunto de credenciais antes que o antigo expire. Em alguns SDKs, você pode usar um provedor que gerencie o processo de atualização de credenciais para você; verifique a documentação do SDK que você está usando.

Uso de credenciais de segurança temporárias com a AWS CLI

Você pode usar credenciais de segurança temporárias com a AWS CLI. Isso pode ser útil para testar políticas.

Usando a AWS CLI, você pode chamar uma API do AWS STS como `AssumeRole` ou `GetFederationToken` e, depois capturar os resultados gerados. O exemplo a seguir mostra uma chamada para `AssumeRole` que envia a saída para um arquivo. No exemplo, o parâmetro `profile` é considerado um perfil no arquivo de configuração da AWS CLI. Ele também é considerado como referência às credenciais de um usuário do IAM que tem permissões para assumir a função.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Quando o comando for concluído, você poderá extrair o ID de chave de acesso, a chave de acesso secreta e o token de sessão de onde quer que você o tenha roteado. Você pode fazer isso manualmente ou usando um script. Em seguida, você pode atribuir esses valores a variáveis do ambiente.

Quando você executa comandos da AWS CLI, a AWS CLI procura as credenciais em uma ordem específica – primeiro em variáveis do ambiente e, em seguida, no arquivo de configuração. Portanto, depois de colocar as credenciais temporárias em variáveis do ambiente, a AWS CLI usa essas credenciais por padrão. (Se você especificar um parâmetro `profile` no comando, a AWS CLI ignorará as variáveis de ambiente. Em vez disso, a AWS CLI procurará no arquivo de configuração, que permite substituir as credenciais nas variáveis de ambiente, se necessário.)

O exemplo a seguir mostra como definir as variáveis do ambiente para credenciais de segurança temporárias e, em seguida, chamar um comando da AWS CLI. Como nenhum parâmetro `profile` está incluído no comando da AWS CLI, a AWS CLI procura credenciais primeiro em variáveis do ambiente e depois usa as credenciais temporárias.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Uso de credenciais de segurança temporárias com operações de API

Se estiver fazendo solicitações de API de HTTPS diretas à AWS, será possível assinar essas solicitações com as credenciais de segurança temporárias obtidas do AWS Security Token Service (AWS STS). Para isso, você pode usar o ID de chave de acesso e a chave de acesso secreta recebidos do AWS STS. Use o ID da chave de acesso e a chave de acesso secreta do mesmo modo que você usaria credenciais de longo prazo para assinar uma solicitação. Você também adiciona à sua solicitação de API o token de sessão que você recebe do AWS STS. Você adiciona o token de sessão a um cabeçalho HTTP ou a um parâmetro de string de consulta denominado `X-Amz-Security-Token`. Você adiciona o token de sessão ao cabeçalho HTTP ou ao parâmetro de string de consulta, mas não a ambos. Para obter mais informações sobre como assinar solicitações da API HTTPS, consulte Como assinar solicitações de API da AWS na Referência geral da AWS.

Mais informações

Para obter mais informações sobre como usar o AWS STS com outros produtos da AWS, consulte os seguintes links:

- Amazon S3. Consulte Fazer solicitações usando credenciais temporárias de usuário do IAM ou Fazer solicitações usando credenciais temporárias de usuário federado, no Guia do usuário do Amazon Simple Storage Service.
- Amazon SNS. Consulte Usar políticas baseadas em identidade com o Amazon SNS, no Guia do desenvolvedor do Amazon Simple Notification Service.
- Amazon SQS. Consulte Gerenciamento de identidade e acesso no Amazon SQS no Guia do desenvolvedor do Amazon Simple Queue Service.
- Amazon SimpleDB. Consulte Usar credenciais de segurança temporárias no Guia do desenvolvedor do Amazon SimpleDB.

Permissões de credenciais de segurança temporárias

É possível usar o AWS Security Token Service (AWS STS) para criar e fornecer aos usuários confiáveis credenciais de segurança temporárias que podem controlar o acesso aos seus recursos da AWS. Para obter mais informações sobre o AWS STS, consulte Credenciais de segurança temporárias no IAM. Assim que o AWS STS emite credenciais de segurança temporárias, elas são válidas durante o período de expiração e não podem ser revogadas. No entanto, as permissões atribuídas a credenciais de segurança temporárias são avaliadas todas as vezes que uma solicitação é feita usando as credenciais, portanto você pode atingir o resultado da revogação de credenciais alterando seus direitos de acesso depois que elas forem emitidas.

Os tópicos a seguir presumem que você tenha conhecimento das permissões e políticas da AWS. Para mais informações sobre esses tópicos, consulte Gerenciamento de acesso para recursos da AWS.

Tópicos

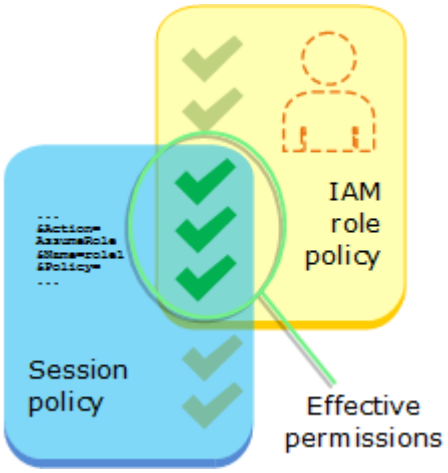
- Permissões para AssumeRole, AssumeRoleWithSAML e AssumeRoleWithWebIdentity
- Monitorar e controlar ações realizadas com funções assumidas
- Permissões para GetFederationToken
- Permissões para GetSessionToken
- Desabilitar permissões de credenciais de segurança temporárias

- Conceder permissões para criar credenciais de segurança temporárias
- Concessão de permissões para o uso de sessões de console aprimoradas com identidade

Permissões para AssumeRole, AssumeRoleWithSAML e AssumeRoleWithWebIdentity

A política de permissões da função que está sendo assumida determina as permissões para as credenciais de segurança temporárias retornadas por `AssumeRole`, `AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity`. Você define essas permissões quando cria ou atualiza uma função.

Você também pode transmitir as políticas de sessão gerenciadas ou em linha como parâmetros das operações de API `AssumeRole`, `AssumeRoleWithWebIdentity` ou `AssumeRoleWithSAML`. As políticas de sessão limitam as permissões para a sessão de credencial temporária da função. As permissões da sessão resultante são a interseção da política baseada em identidade da função e das políticas de sessão. Você pode usar as credenciais temporárias da função em chamadas subsequentes à API da AWS para acessar recursos na conta que possui a função. Você não pode usar políticas de sessão para conceder mais permissões do que as permitidas pela política baseada em identidade da função que está sendo assumida. Para saber mais sobre como a AWS determina as permissões efetivas de uma função, consulte Lógica da avaliação de política.



As políticas que são anexadas às credenciais que fizeram a chamada original para o `AssumeRole` não são avaliadas pela AWS ao tomar a decisão de autorização "permitir" ou "negar". O usuário fornece temporariamente suas permissões originais em favor das permissões atribuídas pelo função assumida. No caso das operações de API `AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity`, não há políticas para avaliar porque o chamador da API não é uma identidade da AWS.

Exemplo: atribuição de permissões usando AssumeRole

Você pode usar uma operação de API `AssumeRole` com diferentes tipos de políticas. Veja a seguir alguns exemplos.

Política de permissões da função

Neste exemplo, você chama a operação de API `AssumeRole` sem especificar a política da sessão no parâmetro `Policy` opcional. As permissões atribuídas às credenciais temporárias são determinadas pela política de permissões da função que está sendo assumida. O exemplo a seguir de política de permissões concede à função permissão para listar todos os objetos contidos em um bucket do S3 chamado `productionapp`. Ele também permite que a função obtenha, adicione e exclua objetos dentro desse bucket.

Example Exemplo de política de permissões da função

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Política de sessão transmitida como parâmetro

Imagine que você deseja permitir que um usuário assuma a mesma função do exemplo anterior. No entanto, neste caso, você deseja que a sessão de função tenha permissão apenas para obter e colocar objetos no bucket `productionapp` do S3. Você não deseja permitir que ele exclua objetos. Uma forma de conseguir isso é criar uma nova função e especificar as permissões desejadas nessa política de permissão da função. Outra forma de fazer isso é chamar a API `AssumeRole` e incluir políticas de sessão no parâmetro opcional `Policy` como parte da operação de API. As permissões da sessão resultante são a interseção das políticas baseadas em identidade da função e das políticas de sessão. As políticas de sessão não podem ser usadas para conceder mais permissões do que as permitidas pela política baseada em identidade da função que está sendo assumida. Para obter mais informações sobre as permissões de sessão da função, consulte Políticas de sessão.

Depois de recuperar as credenciais temporárias da nova sessão, você pode transmiti-las para o usuário que você deseja que tenha essas permissões.

Por exemplo, imagine que a seguinte política é transmitida como um parâmetro da chamada de API. A pessoa que usa a sessão tem permissões para executar apenas as seguintes ações:

- Listar todos os objetos no bucket `productionapp`.
- Obter e colocar objetos no bucket `productionapp`.

Na política de sessão a seguir, a permissão `s3:DeleteObject` é filtrada e a sessão assumida não recebe a permissão `s3:DeleteObject`. A política define o máximo de permissões para a sessão da função de forma que ele substitui todas as políticas de permissões existentes na função.

Example Exemplo de política de sessão passada com a chamada da API **`AssumeRole`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Política baseada em recurso

Alguns recursos da AWS dão suporte às políticas baseadas em recursos, e essas políticas fornecem outro mecanismo para definir permissões que afetam credenciais de segurança temporárias. Apenas alguns recursos, como buckets do Amazon S3, tópicos do Amazon SNS e filas do Amazon SQS oferecem suporte a políticas baseadas em recurso. O exemplo a seguir expande os exemplos anteriores usando um bucket do S3 denominado `productionapp`. A política a seguir é anexada ao bucket.

Quando você anexa a seguinte política baseada em recurso ao bucket `productionapp`, todos os usuários ficam impedidos de excluir objetos do bucket. (Consulte o elemento `Principal` na política). Isso inclui todos os usuários da função assumida, mesmo que a política de permissões da função conceda a permissão `DeleteObject`. Uma declaração explicita `Deny` sempre tem precedência sobre uma instrução `Allow`.

Example Exemplo de política de bucket

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Para obter mais informações sobre como vários tipos de políticas são combinados e avaliados pela AWS, consulte Lógica da avaliação de política.

Monitorar e controlar ações realizadas com funções assumidas

Uma função do IAM é um objeto no IAM ao qual são atribuídas permissões. Ao assumir essa função usando uma identidade do IAM ou uma identidade de fora da AWS, você recebe uma sessão com as permissões atribuídas à função.

Quando você executa ações na AWS, as informações sobre sua sessão podem ser registradas no AWS CloudTrail para o administrador da conta monitorar. Os administradores podem configurar funções para exigir que as identidades passem uma string personalizada que identifica a pessoa ou a aplicação que está executando ações na AWS. Essas informações de identidade são armazenadas como a identidade-fonte no AWS CloudTrail. Quando o administrador revisa a atividade no CloudTrail, ele pode visualizar as informações de identidade-fonte para determinar quem ou o que executou ações com sessões de funções assumidas.

Depois que uma identidade-fonte é definida, ela estará presente em solicitações para qualquer ação da AWS realizada durante a sessão de função. O valor definido persiste quando uma função é usada para assumir outra função por meio da AWS CLI ou da API da AWS, o que é conhecido como encadeamento de funções. O valor definido não pode ser alterado durante a sessão da função.

Os administradores podem configurar permissões detalhadas com base na presença ou no valor da identidade-fonte para controlar ainda mais as ações da AWS que são tomadas com funções compartilhadas. Você pode decidir se o atributo de identidade-fonte pode ser usado, se é necessário e qual valor pode ser usado.

A maneira como você usa a identidade-fonte difere do nome da sessão de função e das etiquetas de sessão de uma maneira importante. O valor da identidade-fonte não pode ser alterado depois de definido, e ele persiste para quaisquer ações adicionais realizadas com a sessão de função. Veja como você pode usar etiquetas de sessão e nome de sessão de função:

- Etiquetas de sessão: você também pode passar etiquetas de sessão ao assumir uma função ou federar um usuário. As etiquetas de sessão estão presentes quando uma função é assumida. Você pode ainda definir políticas que usam chaves de condição de tag para conceder permissões aos seus principais com base nas tags. Em seguida, você pode usar o CloudTrail para visualizar as solicitações feitas para assumir funções ou federar usuários. Para saber mais sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

- Nome da sessão de função: você pode usar a chave de condição `sts:RoleSessionName` em uma política de confiança de função para exigir que seus usuários forneçam um nome de sessão específico quando assumirem uma função. O nome da sessão de função pode ser usado para diferenciar sessões de função quando uma função é usada por diferentes entidades de segurança. Para saber mais sobre o nome da sessão de função, consulte sts:RoleSessionName.

Recomendamos que você use a identidade-fonte quando quiser controlar a identidade que assume uma função. A identidade-fonte também é útil para mineração de logs do CloudTrail para determinar quem usou a função para executar ações.

Tópicos

- Configuração para usar a identidade-fonte
- O que é preciso saber sobre a identidade-fonte
- Permissões necessárias para definir a identidade-fonte
- Especificação de uma identidade-fonte ao assumir uma função
- Uso da identidade-fonte com AssumeRole
- Uso da identidade-fonte com AssumeRoleWithSAML
- Uso da identidade-fonte com AssumeRoleWithWebIdentity
- Controlar o acesso usando informações de identidade-fonte
- Visualização de uma identidade-fonte no CloudTrail

Configuração para usar a identidade-fonte

A maneira que você configura para usar a identidade-fonte depende do método usado quando suas funções são assumidas. Por exemplo, seus usuários do IAM podem assumir funções diretamente usando a operação `AssumeRole`. Se você tiver identidades empresariais, também conhecidas como identidades do quadro de funcionários, elas poderão acessar seus recursos da AWS usando `AssumeRoleWithSAML`. Se os usuários finais acessarem suas aplicações móveis ou Web, eles poderão fazer isso usando `AssumeRoleWithWebIdentity`. Veja a seguir uma visão geral do fluxo de trabalho de alto nível para ajudar você a entender como configurar para uso as informações de identidade-fonte em seu ambiente existente.

1. Configure usuários e funções de teste: usando um ambiente de pré-produção, configure usuários e funções de teste e configure suas respectivas políticas para permitir a definição de uma identidade-fonte.

Se você usar um provedor de identidade (IdP) para suas identidades federadas, configure seu IdP para passar um atributo de usuário de sua escolha para a identidade-fonte na asserção ou no token.

2. Assuma a função: teste assumindo funções e passando uma identidade-fonte com os usuários e funções que você configurou para teste.
3. Reveja o CloudTrail: reveja as informações de identidade-fonte para suas funções de teste nos logs do CloudTrail.
4. Treine seus usuários: depois de testar em seu ambiente de pré-produção, certifique-se de que seus usuários saibam como transmitir as informações de identidade-fonte, se necessário. Defina um prazo para quando você exigirá que seus usuários forneçam uma identidade-fonte em seu ambiente de produção.
5. Configure políticas de produção: configure as políticas para o ambiente de produção e, em seguida, adicione-as aos usuários e funções de produção.
6. Monitore a atividade: monitore sua atividade de função de produção usando logs do CloudTrail.

O que é preciso saber sobre a identidade-fonte

Lembre-se do seguinte ao trabalhar com a identidade-fonte.

- As políticas de confiança para todas as funções conectadas a um provedor de identidade (IdP) devem ter a permissão `sts:SetSourceIdentity`. Para funções que não têm essa permissão na política de confiança de função, a operação `AssumeRole*` falhará. Se não quiser atualizar a política de confiança de função para cada função, você pode usar uma instância do IdP separada para passar a identidade-fonte. Em seguida, adicione a `sts:SetSourceIdentity` permissão apenas às funções que estiverem conectadas ao IdP separado.
- Quando uma identidade define uma identidade-fonte, a chave `sts:SourceIdentity` fica presente na solicitação. Para ações subsequentes realizadas durante a sessão de função, a chave `aws:SourceIdentity` estará presente na solicitação. A AWS não controla o valor da identidade-fonte nas chaves `sts:SourceIdentity` ou `aws:SourceIdentity`. Se você optar por exigir uma identidade-fonte, deverá escolher um atributo que deseja que seus usuários ou o IdP forneçam. Por motivos de segurança, você deve garantir que pode controlar como esses valores são fornecidos.
- O valor da identidade-fonte deve ter entre 2 e 256 caracteres, pode conter apenas caracteres alfanuméricos, sublinhados e os seguintes caracteres: . , + = @ - (hífen). Você não pode usar um valor que comece com o texto **`aws:`**. Este prefixo está reservado para uso interno da AWS.

- As informações de identidade-fonte não são capturadas pelo CloudTrail quando um produto da AWS ou uma função vinculada ao serviço executa uma ação em nome de uma identidade do quadro de funcionários ou federada.

> **Important**
>
> Você não pode alternar para uma função no Console de gerenciamento da AWS ue exija que uma identidade-fonte seja definida quando a função for assumida. Para assumir essa função, você pode usar a AWS CLI ou a API da AWS para chamar a operação `AssumeRole` e especificar o parâmetro da identidade-fonte.

Permissões necessárias para definir a identidade-fonte

Além da ação que corresponde à operação da API, é necessário ter a seguinte ação somente com permissão em sua política:

```
sts:SetSourceIdentity
```

- Para especificar uma identidade-fonte, as entidades de segurança (usuários e funções do IAM) devem ter permissões para `sts:SetSourceIdentity`. Como administrador, você pode configurar isso na política de confiança da função e na política de permissões da entidade de segurança.
- Quando você assume uma função com outra função, chamada de encadeamento de funções, as permissões para `sts:SetSourceIdentity` são necessárias na política de permissões da entidade de segurança que está assumindo a função e na política de confiança da função de destino. Caso contrário, a operação de assumir função falhará.
- Ao usar a identidade-fonte, as políticas de confiança de função para todas as funções conectadas a um IdP devem ter a permissão `sts:SetSourceIdentity`. A operação `AssumeRole*` falhará para qualquer função conectada a um IdP sem essa permissão. Se você não quiser atualizar a política de confiança de função para cada função, use uma instância de IdP separada para passar a identidade-fonte e adicionar a permissão `sts:SetSourceIdentity` apenas às funções que estão conectadas ao IdP separado.
- Para definir uma identidade-fonte entre os limites da conta, você deve incluir a permissão `sts:SetSourceIdentity` em dois lugares. Ela deve estar na política de permissões da entidade de segurança na conta originadora e na política de confiança da função na conta de destino.

Talvez seja necessário fazer isso, por exemplo, quando uma função for usada para assumir uma função em outra conta com oencadeamento de funções.

Como administrador da conta, imagine que você deseja permitir que o usuário do IAM `DevUser` em sua conta assuma a `Developer_Role` na mesma conta. Mas você deseja permitir essa ação somente se o usuário tiver definido a identidade-fonte como seu próprio nome de usuário do IAM. Você pode anexar a política a seguir ao usuário do IAM.

Example Exemplo de política baseada em identidade anexada ao DevUser

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
}
```

Para impor os valores de identidade-fonte aceitáveis, você pode configurar a política de confiança de função a seguir. A política fornece ao usuário do IAM `DevUser` permissões para assumir a função e definir uma identidade-fonte. A chave de condição `sts:SourceIdentity` define o valor da identidade-fonte aceitável.

Example Exemplo de política de confiança de função para identidade-fonte

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
    }
  ]
}
```

Usando as credenciais `DevUser` do usuário do IAM, o usuário tenta assumir a função `DeveloperRole` usando a solicitação da AWS CLI a seguir.

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Quando a AWS avalia a solicitação, o contexto da solicitação contém a `sts:SourceIdentity` de `DevUser`.

Especificação de uma identidade-fonte ao assumir uma função

Você pode especificar uma identidade-fonte ao usar uma das operações da API `AssumeRole*` do AWS STS para obter credenciais de segurança temporárias para uma função. A operação de API que você usa difere dependendo do seu caso de uso.

Por exemplo, se você usar funções do IAM para dar aos usuários do IAM acesso aos recursos da AWS quais eles normalmente não têm acesso, você poderá usar a operação `AssumeRole`. Se você usar a federação de identidade empresarial para gerenciar os usuários do quadro de funcionários, poderá usar a operação `AssumeRoleWithSAML`. Se você usar a federação OIDC para permitir que os usuários finais acessem suas aplicações móveis ou Web, use a operação `AssumeRoleWithWebIdentity`. As seções a seguir explicam como usar a identidade-fonte em cada operação. Para saber mais sobre cenários comuns de credenciais temporárias, consulte Cenários comuns para credenciais temporárias.

Uso da identidade-fonte com AssumeRole

A operação `AssumeRole` retorna um conjunto de credenciais temporárias que você pode usar para acessar recursos da AWS. Você pode usar o usuário do IAM ou credenciais de função para chamar `AssumeRole`. Para passar a identidade-fonte enquanto assume uma função, use a opção `--source-identity` da AWS CLI ou o parâmetro `SourceIdentity` da API da AWS. O exemplo a seguir mostra como especificar a identidade-fonte usando a AWS CLI.

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

Uso da identidade-fonte com AssumeRoleWithSAML

A entidade de segurança que chama a operação `AssumeRoleWithSAML` é autenticada usando a federação baseada em SAML. Essa operação retorna um conjunto de credenciais temporárias que você pode usar para acessar os recursos da AWS. Para obter mais informações sobre como usar a federação baseada em SAML para acesso ao Console de gerenciamento da AWS, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS. Para obter detalhes sobre acesso à AWS CLI ou à API da AWS, consulte Federação SAML 2.0.

Para obter um tutorial sobre como configurar a federação do SAML para seus usuários do Active Directory, consulte AWS Federated Authentication with Active Directory Federation Services (ADFS) no AWS Security Blog.

Como administrador, você pode permitir que os membros do diretório da empresa se agrupem na AWS usando a operação AWS STS da `AssumeRoleWithSAML`. Para isso, é necessário concluir as seguintes tarefas:

1. Configure um provedor SAML na sua organização.
2. Criar um provedor SAML no IAM.
3. Configurar um perfil e suas permissões na AWS para suas entidades principais federadas SAML.
4. Concluir a configuração do IdP SAML e criar declarações para a resposta de autenticação SAML.

Para definir um atributo SAML para a identidade-fonte, inclua o elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Use o elemento `AttributeValue` para especificar o valor da identidade-fonte. Por exemplo, suponha que você deseja passar o seguinte atributo de identidade como a identidade-fonte.

```
SourceIdentity:DiegoRamirez
```

Para passar esse atributo, inclua o seguinte elemento em sua declaração SAML.

Example Exemplo de trecho de uma declaração SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Uso da identidade-fonte com AssumeRoleWithWebIdentity

A entidade principal que chama a operação `AssumeRoleWithWebIdentity` é autenticada usando federação compatível com o OpenID Connect (OIDC). Essa operação retorna um conjunto de credenciais temporárias que você pode usar para acessar os recursos da AWS. Para obter mais informações sobre como usar a federação OIDC para acesso ao Console de gerenciamento da AWS, consulte Federação OIDC.

Para passar a identidade-fonte do OpenID Connect (OIDC), é necessário incluir a identidade-fonte no JSON Web Token (JWT). Inclua a identidade-fonte no namespace `https://aws.amazon.com/` `source_identity` no token ao enviar a solicitação

`AssumeRoleWithWebIdentity`. Para saber mais sobre tokens e reivindicações OIDC, consulte Usar tokens com grupos de usuários no Guia do desenvolvedor Amazon Cognito.

Por exemplo, o JWT decodificado a seguir é um token usado para chamar `AssumeRoleWithWebIdentity` com a identidade-fonte`Admin`.

Example Exemplo de JSON Web Token decodificado

```
{
    "sub": "john",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Controlar o acesso usando informações de identidade-fonte

Quando uma identidade-fonte é definida inicialmente, a chave sts:SourceIdentity fica presente na solicitação. Depois que uma identidade-fonte for definida, a chave aws:SourceIdentity estará presente em todas as solicitações subsequentes feitas durante a sessão de função. Como administrador, você pode escrever políticas que concedem autorização condicional para realizar ações da AWS com base na existência ou no valor do atributo de identidade-fonte.

Imagine que você deseja exigir que seus desenvolvedores definam uma identidade-fonte para assumir uma função crítica que tenha permissão para gravar em um recurso crítico de produção da AWS. Imagine também que você conceda acesso da AWS às identidades do quadro de funcionários usando `AssumeRoleWithSAML`. Você deseja que apenas os desenvolvedores sênior Saanvi e Diego tenham acesso à função, portanto, você cria a seguinte política de confiança para a função.

Example Exemplo de política de confiança de função para identidade-fonte (SAML)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://region-code.signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

A política de confiança possui uma condição para `sts:SourceIdentity` que requer uma identidade-fonte definida como Saanvi ou Diego para assumir a função crítica.

Como alternativa, se você usar um provedor OIDC para federação e os usuários forem autenticados com `AssumeRoleWithWebIdentity`, sua política de confiança de perfil poderá ser semelhante à mostrada a seguir.

Example Exemplo de política de confiança de função para identidade-fonte (provedor OIDC)

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

Requisitos de encadeamento de funções e entre contas

Imagine que você deseja permitir que os usuários que assumiram a `CriticalRole` assumam uma `CriticalRole_2` em outra conta. As credenciais da sessão de função que foram obtidas para assumir `CriticalRole` são usadas para encadear funções para uma segunda função, `CriticalRole_2`, em outra conta. A função está sendo assumida além dos limites de uma conta.

Portanto, a permissão `sts:SetSourceIdentity` deve ser concedida na política de permissões em `CriticalRole` e na política de confiança de função em `CriticalRole_2`.

Example Exemplo de política de permissões em CriticalRole

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Para proteger a definição de identidade-fonte além do limite da conta, a seguinte política de confiança de função confia apenas na entidade de segurança da função de `CriticalRole` para definir a identidade-fonte.

Example Exemplo de política de confiança de função em CriticalRole_2

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
```

```
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

O usuário faz a seguinte chamada usando credenciais de sessão de função obtidas assumindo a CriticalRole. A identidade-fonte foi definida durante a suposição de CriticalRole, portanto, ela não precisa ser explicitamente definida novamente. Se o usuário tentar definir uma identidade-fonte diferente do valor definido quando `CriticalRole` foi assumida, a solicitação de assumir função será negada.

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Quando a entidade de segurança de chamada assume a função, a identidade-fonte na solicitação persiste desde a primeira sessão de função assumida. Portanto, as chaves `aws:SourceIdentity` e `sts:SourceIdentity` estão presentes no contexto da solicitação.

Visualização de uma identidade-fonte no CloudTrail

Você pode usar o CloudTrail para visualizar as solicitações feitas para assumir funções ou federar usuários. Você também pode visualizar as solicitações de função ou do usuário para realizar ações na AWS. O arquivo de log do CloudTrail inclui informações sobre a identidade-fonte definida para a função assumida ou a sessão de usuário federado. Para obter mais informações, consulte . Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail

Por exemplo, suponha que um usuário faça uma solicitção `AssumeRole` ao AWS STS e defina uma identidade-fonte. Você pode encontrar as informações de `sourceIdentity` na chave `requestParameters` em seu log do CloudTrail.

Example Exemplo de seção requestParameters em um log do AWS CloudTrail

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Se o usuário usar a sessão de função assumida para executar uma ação, as informações da identidade-fonte estarão presentes na chave `userIdentity` no log do CloudTrail.

Example Exemplo de chave userIdentity em um log do AWS CloudTrail

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
```

```
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

Para ver exemplos de eventos de API do AWS STS em logs do CloudTrail, consulte Exemplo de eventos de API do IAM no log do CloudTrail. Para obter mais detalhes sobre as informações contidas nos arquivos de log do CloudTrail, consulte Referência de evento do CloudTrail no Guia do usuário do AWS CloudTrail.

Permissões para GetFederationToken

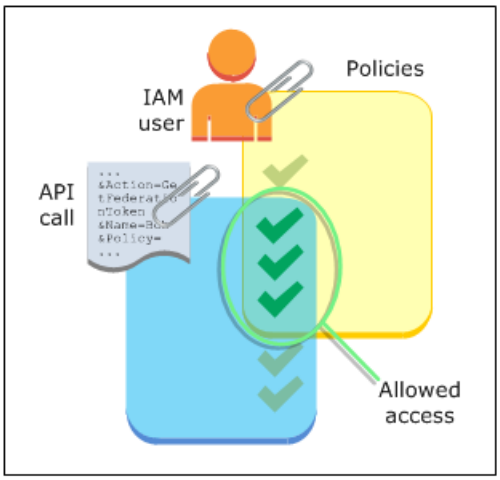
A operação `GetFederationToken` é chamada por um usuário do IAM e retorna credenciais temporárias para esse usuário. Essa operação federa o usuário. As permissões atribuídas a uma sessão de usuário federado do AWS STS são definidas em um de dois lugares:

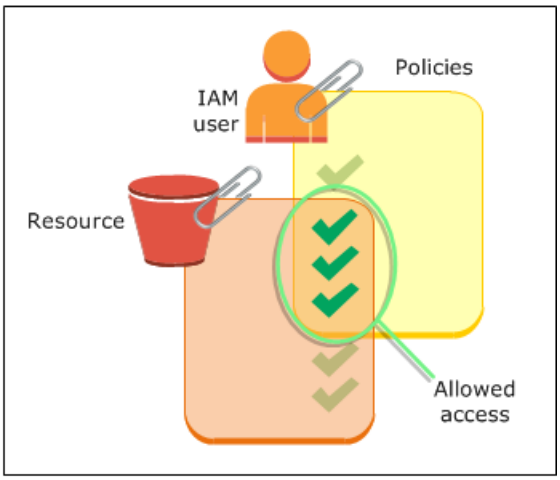
- A política de sessão transmitida como um parâmetro da chamada de API `GetFederationToken`. (Isso é mais comum.)
- Uma política com base em recursos que nomeia explicitamente o usuário federado do AWS STS no elemento `Principal` da política. (Isso é menos comum.)

As políticas de sessão são políticas avançadas que você transmite como parâmetros ao criar de forma programática uma sessão temporária. Quando você cria uma sessão de usuário federado do AWS STS e transmite as políticas de sessão, as permissões da sessão resultante são a interseção da política baseada em identidade do usuário e das políticas da sessão. Você não pode usar a política de sessão para conceder mais permissões do que as permitidas pela política baseada em identidade do usuário que está sendo federado.

Na maioria dos casos, se você não transmitir uma política com a chamada de API `GetFederationToken`, as credenciais de segurança temporárias não terão permissões. No entanto, uma política baseada em recurso pode fornecer permissões adicionais para a sessão. Você pode acessar um recurso com uma política baseada em recurso que especifica sua sessão como a entidade principal permitida.

As figuras a seguir mostram uma representação visual de como as políticas interagem para determinar permissões para as credenciais de segurança temporárias retornadas por uma chamada de `GetFederationToken`.





Exemplo: atribuição de permissões usando GetFederationToken

Você pode usar a ação de API `GetFederationToken` com diferentes tipos de políticas. Veja a seguir alguns exemplos.

Política anexada ao usuário do IAM

Neste exemplo, você tem uma aplicação cliente baseada em navegador que conta com dois serviços da Web de backend. Um serviço de backend é o seu próprio servidor de autenticação que usa o seu próprio sistema de identidade para autenticar a aplicação cliente. O outro serviço de backend é um serviço do AWS que fornece algumas das funcionalidades da aplicação cliente. O aplicativo cliente é autenticado pelo seu servidor, e este cria ou recupera a política de permissões apropriada. Seu servidor chama a API `GetFederationToken` para obter as credenciais de segurança temporárias e retorna essas credenciais para o aplicativo cliente. O aplicativo cliente pode então fazer solicitações diretamente ao serviço da AWS com as credenciais de segurança temporárias. Essa arquitetura permite que o aplicativo cliente faça solicitações da AWS sem incorporação das credenciais da AWS de longo prazo.

Seu servidor de autenticação chama a API `GetFederationToken` com as credenciais de segurança de longo prazo de um usuário do IAM chamado `token-app`. No entanto, as credenciais do usuário do IAM de longo prazo permanecem no servidor e nunca são distribuídas para o cliente. A política do exemplo a seguir está anexada ao usuário `token-app` do IAM e define o mais amplo conjunto de permissões necessários para seus usuários federados do AWS STS (clientes). Observe

que a permissão `sts:GetFederationToken` é necessária para o seu serviço de autenticação para obter credenciais de segurança temporárias para os usuários federados do AWS STS.

Example Exemplo de política anexada ao usuário do IAM **`token-app`** que chama **`GetFederationToken`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

A política anterior concede várias permissões ao usuário do IAM. No entanto, essa política sozinha não concede permissões ao usuário federado do AWS STS. Se esse usuário do IAM chamar

`GetFederationToken` e não transmitir uma política como um parâmetro da chamada de API, o usuário federado do AWS STS resultante não terá permissões efetivas.

Política de sessão transmitida como um parâmetro

A forma mais comum para garantir que o usuário federado do AWS STS tenha a atribuição da permissão apropriada é transmitir políticas de sessão na chamada de API `GetFederationToken`. Expandindo o exemplo anterior, imagine que ação `GetFederationToken` é chamada com as credenciais do usuário do IAM `token-app`. Por exemplo, imagine que a seguinte política de sessão seja transmitida como um parâmetro da chamada de API. O usuário federado do AWS STS resultante tem permissão para listar o conteúdo do bucket do Amazon S3 chamado `productionapp`. O usuário não pode executar as ações do `GetObject` `PutObject` e `DeleteObject` do Amazon S3 em itens no bucket `productionapp`.

Essas permissões são atribuídas ao usuário federado porque as permissões são a interseção das políticas de usuário do IAM e das políticas de sessão que você transmite.

O usuário federado do AWS STS não podia executar ações no Amazon SNS, Amazon SQS, Amazon DynamoDB ou em qualquer bucket do S3, exceto `productionapp`. Essas ações são negadas mesmo que essas permissões sejam concedidas ao usuário do IAM que está associado à chamada `GetFederationToken`.

Example Exemplo de política de sessão passada como parâmetro de chamada de API **`GetFederationToken`**

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
```

```
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Políticas baseadas em recursos

Alguns recursos da AWS oferecem suporte às políticas baseadas em recursos, e essas políticas fornecem outro mecanismo para conceder permissões diretamente ao usuário federado do AWS STS. Apenas alguns serviços da AWS oferecem suporte para políticas baseadas em recursos. Por exemplo, o Amazon S3 tem buckets, o Amazon SNS tem tópicos e o Amazon SQS tem filas às quais você pode anexar políticas. Para obter uma lista de todos os serviços que oferecem suporte às políticas baseadas em recurso, consulte AWSServiços da que funcionam com o IAM e analise a coluna "Políticas baseadas em recurso" das tabelas. Você pode usar as políticas baseadas em recurso para atribuir permissões diretamente a um usuário federado do AWS STS. Faça isso ao especificar o nome do recurso da Amazon (ARN) do usuário federado do AWS STS no elemento `Principal` da política baseada em recurso. O exemplo a seguir ilustra isso e explora mais os exemplos anteriores, usando um bucket do S3 chamado `productionapp`.

A seguinte política baseada em recurso é anexada a um bucket. Essa política do bucket permite que um usuário federado do AWS STS chamado Carol acesse o bucket. Quando a política de exemplo descrita anteriormente é anexada ao usuário do IAM `token-app`, o usuário federado do AWS STS chamado Carol tem permissão para executar as ações `s3:DeleteObject`, `s3:GetObject` e `s3:PutObject` no bucket chamado `productionapp`. Isso é verdadeiro mesmo quando nenhuma política de sessão é transmitida como um parâmetro da chamada de API `GetFederationToken`. Isso porque, nesse caso, a política baseada em recursos a seguir concedeu explicitamente permissões ao usuário federado do AWS STS chamado Carol.

Lembre-se de que um usuário federado do AWS STS recebe permissões apenas quando essas permissões são concedidas explicitamente ao usuário do IAM e ao usuário federado do AWS STS. Elas também podem ser concedidas (dentro de uma conta) por uma política baseada em recurso que nomeie explicitamente o usuário federado do AWS STS no elemento `Principal` da política, como no exemplo a seguir.

Example Exemplo de política de bucket que permite acesso ao usuário federado

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Principal": {
            "AWS": "arn:aws:sts::111122223333:federated-user/Carol"
        },
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:DeleteObject"
        ],
        "Resource": [
            "arn:aws:s3:::productionapp/*"
        ]
    }
}
```

Para obter mais informações sobre como as políticas são avaliadas, consulte Lógica de avaliação da política.

Permissões para GetSessionToken

A principal ocasião para chamar a operação de API `GetSessionToken` ou o comando `get-session-token` da CLI é quando um usuário deve ser autenticado com Multi-Factor Authentication (MFA). É possível gravar uma política que permite determinadas ações somente quando essas ações são solicitadas por um usuário que foi autenticado com o MFA. Para passar na verificação de autorização MFA, um usuário deve primeiro chamar `GetSessionToken` e incluir os parâmetros `SerialNumber` e `TokenCode` opcionais. Se o usuário for autenticado com êxito com um dispositivo MFA, as credenciais retornadas pela operação de API `GetSessionToken` incluirão o contexto de MFA. Esse contexto indica que o usuário é autenticado com a MFA e é autorizado para operações de API que exigem autenticação de MFA.

Permissões necessárias para GetSessionToken

Nenhuma permissão é necessária para que um usuário obtenha um token de sessão. O objetivo da operação `GetSessionToken` é autenticar o usuário que usa a MFA. Não é possível usar políticas para controlar as operações de autenticação.

Para conceder permissões para a execução da maioria das operações da AWS, adicione a ação com o mesmo nome a uma política. Por exemplo, para criar um usuário, você deve usar a operação de API `CreateUser`, o comando `create-user` da CLI ou o Console de gerenciamento da AWS. Para executar essas operações, você deve ter uma política que permite acessar a ação `CreateUser`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateUser",
            "Resource": "*"
        }
    ]
}
```

Você pode incluir a ação `GetSessionToken` em suas políticas, mas não terá efeito sobre a capacidade de um usuário executar a operação `GetSessionToken`.

Permissões concedias por GetSessionToken

Se a ação `GetSessionToken` for chamada com as credenciais de um usuário do IAM, as credenciais de segurança temporárias terão as mesmas permissões que o usuário do IAM. Da mesma forma, se a ação `GetSessionToken` for chamada com credenciais de Usuário raiz da conta da AWS, as credenciais de segurança temporárias terão permissões de usuário raiz.

> **Note**
>
> Recomendamos que você não chame a ação `GetSessionToken` com as credenciais de usuário raiz. Em vez disso, siga nossas práticas recomendadas e crie usuários do IAM com

as permissões de que precisam. Em seguida, use esses usuários do IAM para a interação diária com a AWS.

As credenciais temporárias que você obtém ao chamar `GetSessionToken` têm os seguintes recursos e limitações:

- Você pode usar as credenciais para acessar o Console de gerenciamento da AWS especificando as credenciais para o endpoint de logon único de federação em `https://signin.aws.amazon.com/federation`. Para obter mais informações, consulte Habilitar o acesso do intermediador de identidades personalizado ao console da AWS.
- Você não pode usar as credenciais para chamar as operações de API do IAM ou do AWS STS. Você pode usá-las para chamar operações de API para outros serviços da AWS.

Compare essa operação de API e suas limitações e recursos com as outras operações de API que criam credenciais de segurança temporárias em Compare credenciais do AWS STS

Para obter mais informações sobre o acesso de API protegido por MFA usando `GetSessionToken`, consulte Acesso seguro à API com a MFA.

Desabilitar permissões de credenciais de segurança temporárias

As credenciais de segurança temporárias são válidas até que expirem. Essas credenciais são válidas pela duração especificada, de 900 segundos (15 minutos) até um máximo de 129.600 segundos (36 horas). A duração padrão da sessão é de 43.200 segundos (12 horas). Você pode revogar essas credenciais, mas deverá também alterar as permissões do usuário ou perfil do IAM para impedir o uso de credenciais comprometidas para atividades de conta maliciosas. As permissões atribuídas às credenciais de segurança temporárias são avaliadas cada vez que são usadas para fazer uma solicitação da AWS. Quando você remove todas as permissões das credenciais, as solicitações da AWS que as usam falham.

Pode levar alguns minutos para que as atualizações da política entrem em vigor. Para sessões de perfil do IAM, você pode revogar as credenciais de segurança temporárias do perfil para forçar todos os usuários que assumirem o perfil a se autenticar novamente e a solicitar novas credenciais. Para obter mais informações, consulte Revogar as credenciais de segurança temporárias do perfil.

Não é possível alterar as permissões para um Usuário raiz da conta da AWS. Da mesma forma, você não pode alterar as permissões para as credenciais de segurança temporárias criadas

chamando `GetFederationToken` ou `GetSessionToken` enquanto estiver conectado como usuário raiz. Por esse motivo, recomendamos que você não chame o `GetFederationToken` nem o `GetSessionToken` como usuário raiz.

Para obter os procedimentos sobre como alterar as permissões para um usuário do IAM, consulte Alterar permissões de um usuário do IAM.

Para obter os procedimentos sobre como alterar as permissões para um perfil do IAM, consulte Atualizar permissões para um perfil.

> **Important**
>
> Não é possível editar perfis no IAM que foram criados a partir de conjuntos de permissões do Centro de Identidade do IAM. É necessário revogar a sessão ativa do conjunto de permissões de um usuário no Centro de Identidade do IAM. Para obter mais informações, consulte Revogar sessões ativas de perfil do IAM criadas por conjuntos de permissões, no Guia do usuário do Centro de Identidade do IAM.

Tópicos

- Negar acesso a todas as sessões de perfil do IAM associadas a um perfil
- Negar acesso a uma determinada sessão de perfil do IAM
- Negar acesso a sessões de credenciais de segurança temporárias com chaves de contexto de condição
- Negar acesso a uma determinada entidade principal com políticas baseadas em identidade

Negar acesso a todas as sessões de perfil do IAM associadas a um perfil

Esse procedimento nega permissões para todas as sessões de perfil do IAM associadas a um perfil. Use essa abordagem quando tiver preocupações com relação a acesso suspeito por meio de:

- Entidades principais de outra conta usando acesso entre contas
- Identidades de usuários externos com permissões de acesso a recursos da AWS em sua conta
- Os usuários que foram autenticados em uma aplicação móvel ou Web com um provedor de OIDC

Para alterar ou remover as permissões atribuídas às credenciais de segurança temporárias obtidas chamando as APIs `AssumeRole`, `AssumeRoleWithSAML`, `AssumeRoleWithWebIdentity`, `GetFederationToken` ou `GetSessionToken`, você pode editar ou excluir a política baseada em identidade que define as permissões do perfil.

> **Important**
>
> Se houver uma política baseada em recursos que permita o acesso da entidade principal, você também deverá adicionar uma negação explícita para o recurso. Para mais detalhes, consulte Negar acesso a uma determinada entidade principal com políticas baseadas em identidade.

Para negar acesso a todas as sessões de perfil do IAM associadas a um perfil

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM.
2. No painel de navegação, escolha Perfis.
3. Selecione o nome do perfil a editar. Você pode usar a caixa de pesquisa para filtrar a lista.
4. Escolha a aba Permissões.
5. Selecione a política relevante a editar. Antes de editar uma política gerenciada pelo cliente, revise a guia Entidades anexadas para evitar interromper o acesso a outras identidades que tenham a mesma política anexada.
6. Escolha a guia JSON e atualize a política para negar todos os recursos e ações.

> **Note**
>
> Essas permissões são as mesmas da política gerenciada pela AWS AWSDenyAll. Você pode anexar essa política gerenciada pela AWS a qualquer usuário ou perfil do IAM ao qual você queira negar acesso.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "DenyAll",
            "Effect": "Deny",
            "Action": [
                "*"
            ],
            "Resource": "*"
        }
    ]
}
```

7. Na página Revisão, revise o Resumo da política e, em seguida, selecione Salvar alterações para salvar seu trabalho.

Quando você atualiza a política, as alterações afetam as permissões de todas as credenciais de segurança temporárias associadas ao perfil, incluindo credenciais emitidas antes da alteração da política de permissões do perfil.

Após atualizar a política, você poderá revogar as credenciais de segurança temporárias do perfil para revogar imediatamente todas as permissões às credenciais emitidas pelo perfil.

Negar acesso a uma determinada sessão de perfil do IAM

Quando você atualiza perfis do IAM com uma política de negação total ou exclui o perfil, todos os usuários que têm acesso ao perfil são afetados. Você pode negar acesso sem afetar as permissões de todas as outras sessões associadas ao perfil.

É possível negar permissões à `Principal` usando chaves de contexto de condição ou políticas baseadas em recursos.

> **Tip**
>
> Você pode encontrar os ARNs de usuários federados usando logs do AWS CloudTrail. Para obter mais informações, consulte a página How to Easily Identify Your Federated Users by Using AWS CloudTrail.

Negar acesso a sessões de credenciais de segurança temporárias com chaves de contexto de condição

Você pode usar chaves de contexto de condição nas políticas baseadas em identidade quando desejar negar acesso a sessões específicas de credenciais de segurança temporárias sem afetar

as permissões do usuário ou perfil do IAM que criou as credenciais. Para perfis do IAM, depois de atualizar a política, você também pode revogar as sessões de credenciais de segurança temporárias do perfil para revogar imediatamente todas as credenciais emitidas.

Para obter mais informações sobre chaves de contexto de condição, consulte AWSChaves de contexto de condições globais da.

aws:PrincipalArn

Você pode usar a chave de contexto de condição aws:PrincipalArn em uma política baseada em identidade para negar acesso a uma determinada entidade principal por seu nome do recurso da Amazon (ARN). Para fazer isso, especifique o ARN da sessão do usuário do IAM, do perfil ou do usuário federado do AWS STS à qual as credenciais de segurança temporárias estão associadas no elemento de condição de uma política.

Para negar acesso a uma determinada entidade principal por ARN

1. No painel de navegação do console do IAM, escolha Usuários ou Perfis.
2. Escolha o nome do usuário ou perfil do IAM a editar. Você pode usar a caixa de pesquisa para filtrar a lista.
3. Escolha a aba Permissões.
4. Selecione a política relevante a editar. Antes de editar uma política gerenciada pelo cliente, revise a guia Entidades anexadas para evitar interromper o acesso a outras identidades que tenham a mesma política anexada.
5. Escolha a guia JSON e adicione uma instrução de negação para o ARN da entidade principal, conforme mostrado no exemplo a seguir.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
```

```
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:iam::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

6. Na página Revisão, revise o Resumo da política e, em seguida, selecione Salvar alterações para salvar seu trabalho.

aws:SourceIdentity

Você pode usar a chave de contexto de condição aws:SourceIdentity em uma política baseada em identidade para negar acesso a uma determinada identidade de origem associada a uma sessão de perfil do IAM. Isso se aplica contanto que a sessão de perfil tenha sido emitida definindo o parâmetro de solicitação `SourceIdentity` quando a entidade principal assumiu um perfil usando qualquer comando `assume-role` da CLI do AWS STS* ou operações da API `AssumeRole` do AWS STS*. Faça isso especificando a identidade de origem à qual as credenciais de segurança temporárias estão associadas no elemento `Condition` de uma política.

Ao contrário da chave de contexto sts:RoleSessionName, após a definição da identidade-fonte, o valor não poderá ser alterado. A chave `aws:SourceIdentity` estará presente no contexto da solicitação para todas as ações executadas pelo perfil. A identidade-fonte persiste nas sessões de perfil subsequentes quando você usa as credenciais da sessão para assumir outro perfil. Assumir uma função de outra é chamado de encadeamento de funções.

A política a seguir mostra um exemplo de como é possível negar acesso a sessões temporárias de credenciais de segurança usando a chave de contexto de condição `aws:SourceIdentity`. Se você especificar a identidade-fonte associada a uma sessão de perfil, serão negadas sessões de perfil com a identidade-fonte nomeada sem afetar as permissões da perfil que criou as credenciais. Neste exemplo, a identidade de origem definida pela entidade principal quando a sessão de perfil foi emitida é `nikki_wolf@example.com`. Qualquer solicitação feita por uma sessão de perfil com a identidade-fonte `nikki_wolf@example.com` será negada porque a identidade-fonte está incluída na condição da política e a política Efeito está definida como `Deny`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": [
            "nikki_wolf@example.com",
            "<source identity value>"
          ]
        }
      }
    }
  ]
}
```

aws:userid

Você pode usar a chave de contexto de condição aws:userid em uma política baseada em identidade para negar acesso a todas ou a determinadas sessões de credenciais de segurança temporárias associadas ao usuário ou perfil do IAM. Isso é feito especificando o identificador exclusivo (ID) da sessão do usuário do IAM, do perfil ou do usuário federado do AWS STS à qual as credenciais de segurança temporárias estão associadas no elemento `Condition` de uma política.

A política a seguir mostra um exemplo de como é possível negar acesso a sessões temporárias de credenciais de segurança usando a chave de contexto de condição `aws:userid`.

- `AIDAXUSER1` representa o ID exclusivo de um usuário do IAM. Especificar o ID exclusivo de um usuário do IAM como um valor para a chave de contexto `aws:userid` negará acesso ao usuário do IAM. Isso inclui todas as sessões de credenciais de segurança temporárias que foram criadas chamando a API `GetSessionToken`.
- `AROAXROLE1:*` representa o ID exclusivo de todas as sessões associadas ao perfil do IAM. Especificar o ID exclusivo de um perfil do IAM e um caractere curinga (*) na porção caller-

specified-role-session-name como um valor para a chave de contexto `aws:userid` negará todas as sessões associadas ao perfil.

- `AROAXROLE2:<caller-specified-role-session-name>` representa o ID exclusivo de uma sessão de perfil assumido. Na porção caller-specified-role-session-name do ID exclusivo do perfil assumido, você poderá especificar o nome da sessão do perfil ou um caractere curinga se o operador de condição StringLike for usado. Se você especificar o nome da sessão do perfil, isso negará a sessão do perfil nomeada sem afetar as permissões do perfil que criou as credenciais. Se você especificar um caractere curinga para o nome da sessão do perfil, negará todas as sessões associadas ao perfil.

> **Note**
>
> O nome da sessão de perfil especificada pelo chamador, que faz parte do identificador exclusivo de uma sessão de perfil assumida, pode mudar durante o encadeamento de perfis. O encadeamento de perfis ocorre quando um perfil assume outro perfil. O nome da sessão do perfil é definido usando o parâmetro de solicitação `RoleSessionName` quando a entidade principal assume um perfil usando a operação da API `AssumeRole` do AWS STS.

- `account-id:<federated-user-caller-specified-name>` representa o ID exclusivo de uma sessão de usuário federado do AWS STS. Um usuário do IAM cria essa sessão chamando a API `GetFederationToken`. Especificar o ID exclusivo de uma sessão de usuário federado do AWS STS nega a sessão federada nomeada sem afetar as permissões do usuário do IAM que criou as credenciais.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
```

```
            "AROAXROLE1:*",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "123456789012:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Para exemplos específicos de valores de chave de entidade principal, consulte Valores de chave de principal. Para obter informações sobre os identificadores exclusivos do IAM e como obtê-los, consulte Identificadores exclusivos.

Negar acesso a uma determinada entidade principal com políticas baseadas em identidade

Para restringir o acesso a uma determinada entidade principal com uma política baseada em recurso, você pode usar as chaves de contexto de condição aws:PrincipalArn ou aws:SourceIdentity no elemento `Condition`. Uma política baseada em recurso é uma política de permissão anexada a um recurso e controla quem pode acessar o recurso e quais ações a pessoa pode executar nele.

Quando usar a chave de contexto `aws:PrincipalARN`, especifique o ARN da sessão do usuário do IAM, do perfil ou do usuário federado do AWS STS associada às credenciais de segurança temporárias no elemento de condição de uma política. O exemplo de política a seguir demonstra como usar a chave de contexto `aws:PrincipalARN`em uma política baseada em recurso:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "ArnEquals": {
        "aws:PrincipalArn": [
          "arn:aws:iam::222222222222:role/ROLENAME",
          "arn:aws:iam::222222222222:user/USERNAME",
```

```
          "arn:aws:sts::222222222222:federated-user/USERNAME"
        ]
      }
    }
  }
}
```

Quando usar a chave de contexto `aws:SourceIdentity`, especifique o valor da identidade de origem associada às credenciais de segurança temporárias do perfil no elemento `Condition` de uma política. Isso se aplica contanto que a sessão de perfil tenha sido emitida definindo o parâmetro de solicitação `SourceIdentity` quando a entidade principal assumiu um perfil usando qualquer comando `assume-role` da CLI do AWS STS* ou operações da API `AssumeRole` do AWS STS*. O exemplo a seguir demonstra como usar a chave de contexto `aws:SourceIdentity` em uma política baseada em recurso:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Principal": "*",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {
      "StringLike": {
        "aws:SourceIdentity": [
          "nikki_wolf@example.com",
          "<source identity value>"
        ]
      }
    }
  }
}
```

Se você atualizar apenas a política baseada em identidade para uma entidade principal, ainda será possível executar as ações permitidas na política baseada em recurso, exceto quando essas ações forem explicitamente negadas na política baseada em identidade.

Para negar acesso a uma determinada entidade principal com uma política baseada em identidade

1. Consulte AWSServiços da que funcionam com o IAM para ver se o serviço oferece suporte a políticas baseadas em recursos.
2. Faça login no Console de gerenciamento da AWS e abra o console do serviço. Cada serviço tem uma localização diferente no console para anexar políticas.
3. Edite a política baseada em recurso. Adicione uma instrução de política "Deny" para especificar as informações de identificação da credencial:

   a. No elemento `Principal`, insira o curinga (*). A entidade principal será restringida no elemento `Condition`.

   b. No elemento `Effect`, insira “Deny.”

   c. Em `Action`, insira o namespace do serviço e o nome da ação a ser negada. Para negar todas as ações, use o caractere curinga (*). Por exemplo: `"s3:*"`.

   d. No elemento `Resource`, insira o ARN do recurso-alvo. Por exemplo: `"arn:aws:s3:::amzn-s3-demo-bucket"`.

   e. No elemento `Condition`, especifique a chave de contexto `aws:PrincipalARN` ou `aws:SourceIdentity`.

      Se você usar a chave de contexto `aws:PrincipalARN`, insira o ARN da entidade principal à qual negar acesso.

      Se você usar a chave de contexto `aws:SourceIdentity`, insira o valor da identidade de origem definido na sessão de perfil à qual negar acesso.
4. Salve seu trabalho.

Conceder permissões para criar credenciais de segurança temporárias

Por padrão, os usuários do IAM não têm permissão para criar credenciais de segurança temporárias para sessões de usuários federados do AWS STS e perfis. Você deve usar uma política para fornecer essas permissões aos usuários. Embora você possa conceder permissões diretamente a um usuário, é altamente recomendável que você conceda permissões para um grupo. Isso torna o gerenciamento de permissões muito mais fácil. Quando alguém não precisar mais executar as tarefas associadas às permissões, bastará removê-las do grupo. Se outra pessoa precisa executar essa tarefa, adicione-a ao grupo para conceder as permissões.

Para conceder a um grupo do IAM permissão para criar credenciais de segurança temporárias para sessões de usuários federados do AWS STS ou perfid, anexe uma política que conceda um ou ambos os seguintes privilégios:

- Para entidades principais federadas OIDC e SAML acessarem um perfil do IAM, conceda acesso a `AssumeRole` do AWS STS.
- Para usuários federados do AWS STS que não precisam de um perfil, conceda acesso a `GetFederationToken` do AWS STS.

Para obter mais informações sobre as diferenças entre o `AssumeRole` e `GetFederationToken` operações de API, consulte Solicitar credenciais de segurança temporárias.

Os usuários do IAM também podem chamar `GetSessionToken` para criar credenciais de segurança temporárias. Nenhuma permissão é necessária para um usuário realizar a chamada `GetSessionToken`. O objetivo dessa operação é autenticar o usuário que usa a MFA. Não é possível usar políticas para controlar a autenticação. Isso significa que não é possível impedir que os usuários do IAM chamem `GetSessionToken` para criar credenciais temporárias.

Example Exemplo de política que concede permissão para assumir uma função

O exemplo de política a seguir concede permissão para chamar `AssumeRole` para o perfil `UpdateApp` na Conta da AWS `123123123123`. Quando `AssumeRole` é usado, o usuário (ou o aplicativo) que cria as credenciais de segurança em nome de um usuário federado não pode delegar permissões que já não tenham sido especificadas na política de permissões da função.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Exemplo de política que concede permissão para criar credenciais de segurança temporárias para um usuário federado

No exemplo a seguir a política concede permissões de acesso `GetFederationToken`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Quando você concede a usuários do IAM permissão para criar credenciais de segurança temporárias para usuários federados do AWS STS com `GetFederationToken`, isso permite que eles deleguem suas próprias permissões. Para obter mais informações sobre a delegação de permissões entre usuários do IAM e Contas da AWS, consulte Exemplos de políticas para delegação de acesso. Para obter mais informações sobre o controle de permissões em credenciais de segurança temporárias, consulte Permissões de credenciais de segurança temporárias.

Example Exemplo de política que concede a um usuário permissão limitada para criar credenciais de segurança temporárias para usuários federados

Quando você permite que um usuário do IAM chame `GetFederationToken`, uma prática recomendada é restringir as permissões que esse usuário do IAM pode delegar. Por exemplo, a política a seguir mostra como permitir que um usuário do IAM crie credenciais de segurança temporárias apenas para usuários federados do AWS STS cujos nomes começam com Manager.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Concessão de permissões para o uso de sessões de console aprimoradas com identidade

As sessões de console aprimoradas com identidade possibilitam que os IDs de usuário e de sessão do Centro de Identidade do AWS IAM sejam incluídos nas sessões de console da AWS quando os usuários fizerem login. Por exemplo, o Amazon Q Developer Pro usa as sessões de console aprimoradas com identidade para oferecer uma experiência de serviço personalizada. Para obter mais informações sobre as sessões de console aprimoradas com identidade, consulte Enabling identity-enhanced console sessions no Guia do usuário do Centro de Identidade do AWS IAM. Para obter informações sobre a configuração do Amazon Q Developer, consulte Configurando o Amazon Q Developer no Guia do usuário do Amazon Q Developer.

Para que as sessões de console aprimoradas com identidade estejam disponíveis para um usuário, é necessário usar uma política baseada em identidades para conceder à entidade principal do IAM a permissão `sts:SetContext` no recurso que representa sua própria sessão de console.

> **Important**
>
> Por padrão, os usuários não têm permissão para configurar o contexto em suas sessões de console aprimoradas com identidade. Para permitir isso, você deve conceder a entidade principal do IAM a permissão `sts:SetContext` em uma política baseada em identidade, conforme mostrado no exemplo de política abaixo.

O exemplo de política baseada em identidades, apresentado a seguir, concede a permissão `sts:SetContext` a uma entidade principal do IAM, permitindo que essa entidade defina o contexto da sessão de console aprimorada com identidade para suas próprias sessões no console da AWS. O recurso de política, `arn:aws:sts::`*`account-id`*`:self`, representa a sessão do chamador da AWS. O segmento do `account-id` do ARN pode ser substituído por um caractere curinga * nos

casos em que a mesma política de permissão é implantada em várias contas, como quando essa política é implantada usando os conjuntos de permissões do IAM Identity Center.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:SetContext",
            "Resource": "arn:aws:sts::111122223333:self"
        }
    ]
}
```

Gerenciar o AWS STS em uma Região da AWS

Um endpoint regional é a URL do ponto de entrada em uma região específica para um serviço da Web da AWS. A AWS recomenda o uso dos endpoints regionais do AWS Security Token Service (AWS STS) em vez do endpoint global, para reduzir a latência, incorporar redundância e aumentar a validade dos tokens de sessão.

Embora o endpoint global (legado) do AWS STS, `https://sts.amazonaws.com`, esteja altamente disponível, ele é hospedado em uma única região da AWS, Leste dos EUA (Norte da Virgínia), e, como outros endpoints, não fornece failover automático para endpoints em outras regiões.

- Reduzir a latência: ao fazer suas chamadas do AWS STS para um endpoint geograficamente mais próximo de seus serviços e aplicações, você pode acessar serviços do AWS STS com menor latência e melhores tempos de resposta.
- Incorporar redundância — Você pode limitar os efeitos de uma falha em uma workload a um número limitado de componentes com um escopo previsível de contenção de impactos. O uso de endpoints regionais do AWS STS permite alinhar o escopo de seus componentes com o escopo de seus tokens de sessão. Para obter mais informações sobre esse pilar de confiabilidade, consulte Usar isolamento de falhas para proteger sua workload no AWS Well-Architected Framework.
- Aumentar a validade do token de sessão: tokens de sessão de endpoints regionais do AWS STS são válidos em todas as Regiões da AWS. Tokens de sessão do endpoint global do STS são válidos apenas em Regiões da AWS que são habilitadas por padrão. Se pretende habilitar uma

nova região para sua conta, você pode usar tokens de sessão de endpoints regionais do AWS STS. Se optar por usar o endpoint global, você deverá alterar a compatibilidade de regiões de tokens de sessão do AWS STS para o endpoint global. Isso garante que os tokens sejam válidos em todas as Regiões da AWS.

Para obter uma lista de regiões da AWS STS e seus endpoints, consulte AWS STSRegiões e endpoints do.

> **Note**
>
> A AWS fez alterações no endpoint global (`https://sts.amazonaws.com`) do AWS Security Token Service (AWS STS) nas regiões habilitadas por padrão para aprimorar sua resiliência e desempenho. As solicitações do AWS STS para o endpoint global são atendidas automaticamente da mesma Região da AWS que suas workloads. Essas alterações não serão implantadas em regiões opt-in. Recomendamos usar endpoints regionais do AWS STS apropriados. Para obter mais informações, consulte Alterações nos endpoints globais do AWS STS.

Tópicos

- Ativar e desativar o AWS STS em uma Região da AWS
- Escrever código para usar regiões do AWS STS
- Gerenciar tokens de sessão de endpoint global

Ativar e desativar o AWS STS em uma Região da AWS

Quando você ativar endpoints do AWS STS para uma região, o AWS STS poderá emitir credenciais temporárias para usuários e funções em sua conta que faz uma solicitação do AWS STS. Essas credenciais podem ser usadas em qualquer região habilitada por padrão ou habilitada manualmente.

Para regiões que são habilitadas por padrão, é necessário ativar o endpoint do AWS STS regional na conta em que as credenciais temporárias são geradas. Não importa se um usuário está conectado na mesma conta ou em uma conta diferente quando fizer a solicitação.

Ao solicitar credenciais temporárias para uma função em outra Conta da AWS usando uma região ativada manualmente, a conta de destino (a conta que contém a função) deve habilitar essa

região para operações AWS STS. Esta é uma forma de garantir que as credenciais de segurança temporárias possam ser geradas corretamente.

Por exemplo, imagine que um usuário na conta A deseje enviar uma solicitação de API `sts:AssumeRole` ao endpoint regional do AWS STS `https://sts.ap-southeast-3.amazonaws.com`. A solicitação é para credenciais temporárias para a função denominada `Developer` na conta B. Como a solicitação é para criar credenciais para uma entidade na conta B, a conta B deve ter a região `ap-southeast-3` ativada. Os usuários da conta A (ou de qualquer outra conta) podem chamar o endpoint do AWS STS `ap-southeast-3` para solicitar credenciais para a conta B, esteja ou não ativada a região em suas contas. Para saber mais, consulte Ativar ou desativar Regiões da AWS em sua conta.

> **Note**
>
> Regiões ativas estão disponíveis para todas as pessoas que usam credenciais temporárias nessa conta. Para controlar quais usuários ou perfis do IAM podem acessar a região, use a chave de condição `aws:RequestedRegion` em suas políticas de permissões.

Para ativar ou desativar o AWS STS em uma região habilitada por padrão (console)

1. Faça login como um usuário raiz ou um usuário com permissões para realizar tarefas de administração do IAM.
2. Abra o console do IAM e, no painel de navegação, escolha Account settings (Configurações da conta).
3. Na seção Endpoints do Security Token Service (STS), localize a região que você deseja configurar e escolha Ativa ou Inativa na coluna Status do STS.
4. Na caixa de diálogo que é exibida, escolha Activate (Ativar) ou Deactivate (Desativar).

Para regiões que precisam ser habilitadas, o AWS STS é ativado automaticamente quando você habilita a região. Depois de habilitar uma região, o AWS STS estará sempre ativo nessa região e você não poderá desativá-lo. Para saber como habilitar regiões que estão desabilitadas por padrão, consulte Especificar qual Regiões da AWS sua conta pode usar no Guia de referência do AWS Gerenciamento de contas.

Escrever código para usar regiões do AWS STS

Depois de ativar uma região, você pode direcionar chamadas de API do AWS STS para essa região. O trecho de código Java a seguir demonstra como configurar um objeto `AWSSecurityTokenService` para fazer solicitações para a região Europa (Milão) (eu-south-1).

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STSO recomenda fazer chamadas para um endpoint regional. Para obter informações sobre como habilitar manualmente uma região, consulte Especificar quais Regiões da AWS sua conta pode usar no Guia de referência do AWS Gerenciamento de contas.

No exemplo, a primeira linha cria uma instância de objeto `EndpointConfiguration` chamado `regionEndpointConfig`, passando o URL do endpoint e a Região da AWS como os parâmetros.

Para aprender a configurar endpoints regionais do AWS STS usando uma variável de ambiente para AWS SDKs, consulte Endpoints regionalizados do AWS STS no Guia de referência de AWS SDKs e ferramentas.

Para todas as outras combinações de linguagem e ambiente de programação, consulte a documentação das para o SDK relevante.

Gerenciar tokens de sessão de endpoint global

Por padrão, a maioria das Regiões da AWS está habilitada para operações em todos os Serviços da AWS. Essas regiões são automaticamente habilitadas para uso com o AWS STS. Algumas regiões, como Ásia-Pacífico (Hong Kong), devem ser habilitadas manualmente. Para saber como habilitar e desabilitar Regiões da AWS, consulte Especificar qual Regiões da AWS sua conta pode usar no Guia de referência do AWS Gerenciamento de contas.

Quando você habilitar essas regiões da AWS, elas serão automaticamente habilitadas para uso com o AWS STS. Você não pode ativar o endpoint do AWS STS para uma região que está desabilitada. Tokens de sessão que são válidos em todas as Regiões da AWS incluem mais caracteres do que os tokens que são válidos em regiões habilitadas por padrão. A alteração dessa configuração pode afetar sistemas existentes em que você armazena tokens temporariamente.

Você pode alterar essa configuração usando o Console de gerenciamento da AWS, a AWS CLI, ou a API da AWS.

Para alterar a compatibilidade de regiões dos tokens de sessão para o endpoint global (console)

1. Faça login como um usuário raiz ou um usuário com permissões para realizar tarefas de administração do IAM. Para alterar a compatibilidade de tokens de sessão, você deve ter uma política que permita a ação `iam:SetSecurityTokenServicePreferences`.
2. Abra o console do IAM. No painel de navegação, selecione Configurações da conta.
3. Em Security Token Service (STS), na seção Session Tokens from the STS endpoints (Tokens de sessão dos endpoints do STS). O endpoint global indica `Valid only in Regiões da AWS enabled by default`. Escolha Alterar.
4. Na caixa de diálogo Alterar compatibilidade da região, selecione Todas as Regiões da AWS. Em seguida, escolha Salvar alterações.

> **Note**
>
> Tokens de sessão que são válidos em todas as Região da AWS incluem mais caracteres do que os tokens que são válidos em regiões habilitadas por padrão. A alteração dessa configuração pode afetar sistemas existentes em que você armazena tokens temporariamente.

Para alterar a compatibilidade de regiões de tokens de sessão para o endpoint global (AWS CLI)

Defina a versão do token da sessão. Tokens de versão 1 são válidos somente em Regiões da AWS que estão disponíveis por padrão. Esses tokens não funcionam em regiões habilitadas manualmente, como Ásia-Pacífico (Hong Kong). Tokens de versão 2 são válidos em todas as regiões. No entanto, tokens de versão 2 incluem mais caracteres e podem afetar sistemas em que você armazena tokens temporariamente.

- `aws iam set-security-token-service-preferences`

Para alterar a compatibilidade de regiões de tokens de sessão para o endpoint global (API da AWS)

Defina a versão do token da sessão. Tokens de versão 1 são válidos somente em Regiões da AWS que estão disponíveis por padrão. Esses tokens não funcionam em regiões habilitadas manualmente, como Ásia-Pacífico (Hong Kong). Tokens de versão 2 são válidos em todas as

regiões. No entanto, tokens de versão 2 incluem mais caracteres e podem afetar sistemas em que você armazena tokens temporariamente.

- `SetSecurityTokenServicePreferences`

AWS STSRegiões e endpoints do

> **Note**
>
> A AWS fez alterações no endpoint global (`https://sts.amazonaws.com`) do AWS Security Token Service (AWS STS) nas regiões habilitadas por padrão para aprimorar sua resiliência e desempenho. As solicitações de AWS STS para o endpoint global são atendidas automaticamente da mesma Região da AWS que suas workloads. Essas alterações não serão implantadas em regiões opt-in. Recomendamos usar endpoints regionais do AWS STS apropriados. Para obter mais informações, consulte Alterações nos endpoints globais do AWS STS.

A tabela a seguir lista as regiões e seus endpoints. Ela indica quais são as ativadas por padrão e quais você pode ativar ou desativar.

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
--Global--	sts.amazonaws.com	Sim	No (Não
Leste dos EUA (Ohio)	sts.us-east-2.amazonaws.com	Yes (Sim	Sim
Leste dos EUA (Norte da Virgínia)	sts.us-east-1.amazonaws.com	Sim	No (Não
Oeste dos EUA (N. da Califórnia)	sts.us-west-1.amazonaws.com	Yes (Sim	Sim

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Oeste dos EUA (Oregon)	sts.us-west-2.amazonaws.com	✓ Yes (Sim	✓ Sim
África (Cidade do Cabo)	sts.af-south-1.amazonaws.com	✗ Não[1]	✗ No (Não
Ásia-Pacífico (Hong Kong)	sts.ap-east-1.amazonaws.com	✗ Não[1]	✗ No (Não
Ásia-Pacífico (Hyderabad)	sts.ap-south-2.amazonaws.com	✗ Não[1]	✗ No (Não
Ásia-Pacífico (Jacarta)	sts.ap-southeast-3.amazonaws.com	✗ Não[1]	✗ No (Não
Ásia-Pacífico (Malásia)	sts.ap-southeast-5.amazonaws.com	✗ Não[1]	✗ No (Não
Ásia-Pacífico (Melbourne)	sts.ap-southeast-4.amazonaws.com	✗ Não[1]	✗ No (Não
Ásia-Pacífico (Mumbai)	sts.ap-south-1.amazonaws.com	✓ Yes (Sim	✓ Sim
Ásia-Pacífico (Osaka)	sts.ap-northeast-3.amazonaws.com	✓ Yes (Sim	✓ Sim
Ásia-Pacífico (Seul)	sts.ap-northeast-2.amazonaws.com	✓ Yes (Sim	✓ Sim
Ásia-Pacífico (Singapura)	sts.ap-southeast-1.amazonaws.com	✓ Yes (Sim	✓ Sim
Ásia-Pacífico (Sydney)	sts.ap-southeast-2.amazonaws.com	✓ Yes (Sim	✓ Yes (Sim

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Ásia-Pacífico (Tailândia)	sts.ap-southeast-7.amazonaws.com	⊗ Não[1]	⊗ No (Não
Ásia-Pacífico (Tóquio)	sts.ap-northeast-1.amazonaws.com	⊙ Yes (Sim	⊙ Sim
Canadá (Central)	sts.ca-central-1.amazonaws.com	⊙ Yes (Sim	⊙ Yes (Sim
Oeste do Canadá (Calgary)	sts.ca-west-1.amazonaws.com	⊗ Não[1]	⊗ No (Não)
China (Pequim)	sts.cn-north-1.amazonaws.com.cn	⊙ Sim[2]	⊗ No (Não)
China (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	⊙ Sim[2]	⊙ Sim
Europa (Frankfurt)	sts.eu-central-1.amazonaws.com	⊙ Yes (Sim	⊙ Sim
Europa (Irlanda)	sts.eu-west-1.amazonaws.com	⊙ Yes (Sim	⊙ Sim
Europa (Londres)	sts.eu-west-2.amazonaws.com	⊙ Yes (Sim	⊙ Sim
Europa (Milão)	sts.eu-south-1.amazonaws.com	⊗ Não[1]	⊗ No (Não
Europa (Paris)	sts.eu-west-3.amazonaws.com	⊙ Yes (Sim	⊙ Yes (Sim
Europa (Espanha)	sts.eu-south-2.amazonaws.com	⊗ Não[1]	⊗ No (Não
Europa (Estocolmo)	sts.eu-north-1.amazonaws.com	⊙ Yes (Sim	⊙ Yes (Sim
Europa (Zurique)	sts.eu-central-2.amazonaws.com	⊗ Não[1]	⊗ No (Não
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	⊗ Não[1]	⊗ No (Não
México (Centro)	sts.mx-central-1.amazonaws.com	⊗ Não[1]	⊗ No (Não)

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Oriente Médio (Bahrein)	sts.me-south-1.amazonaws.com	⊗ Não[1]	⊗ No (Não
Oriente Médio (Emirados Árabes Unidos)	sts.me-central-1.amazonaws.com	⊗ Não[1]	⊗ No (Não
América do Sul (São Paulo)	sts.sa-east-1.amazonaws.com	⊙ Yes (Sim	⊙ Sim

[1]Você deve habilitar a região para usá-la. Isso ativa automaticamente o AWS STS. Não é possível ativar ou desativar manualmente o AWS STS nessas regiões.

[2]Para usar a AWS na China, são necessárias uma conta e credenciais específicas da AWS na China.

Alterações nos endpoints globais do AWS STS

A AWS fez alterações no endpoint global (`https://sts.amazonaws.com`) do AWS Security Token Service (AWS STS) em regiões habilitadas por padrão para aprimorar sua resiliência e performance. Anteriormente, todas as solicitações ao endpoint global do AWS STS eram atendidas por uma única Região da AWS Leste dos EUA (Norte da Virgínia). Agora, nas regiões habilitadas por padrão, as solicitações para o endpoint global do AWS STS serão atendidas automaticamente na mesma região de origem da solicitação, em vez da região Leste dos EUA (Norte da Virgínia). Essas alterações não serão implantadas em regiões opt-in.

Com essas alterações, o AWS STS processará sua solicitação com base na região de origem e no resolvedor de DNS usado. As solicitações ao endpoint global do AWS STS são atendidas na mesma região da sua workload implantada da AWS se a solicitação de DNS ao endpoint global do AWS STS for tratada pelo servidor Amazon DNS em regiões habilitadas por padrão. As solicitações ao endpoint global do AWS STS continuarão sendo atendidas na região Leste dos EUA (Norte da Virgínia) se sua solicitação tiver sido originada de regiões opcionais ou se sua solicitação tiver sido resolvida usando um resolvedor de DNS diferente do servidor Amazon DNS. Para obter mais informações sobre o Amazon DNS, consulte Servidor Amazon DNS no Guia do usuário da Amazon Virtual Private Cloud.

A tabela a seguir mostra como as solicitações ao endpoint global do AWS STS são roteadas com base em seu provedor de DNS.

Resolvedor de DNS	Solicitações ao endpoint global do AWS STS roteadas para a Região da AWS local?
Resolvedor do Amazon DNS em uma Amazon VPC em uma região habilitada por padrão	Sim
Resolvedor do Amazon DNS em uma Amazon VPC em uma região opt-in	Não, a solicitação será encaminhada para a região Leste dos EUA (Norte da Virgínia)
Resolvedor de DNS fornecido pelo seu ISP, um provedor de DNS público ou qualquer outro provedor de DNS	Não, a solicitação será encaminhada para a região Leste dos EUA (Norte da Virgínia)

Para garantir o mínimo de interrupção em seus processos existentes, a AWS implementou as seguintes medidas:

- Os logs do AWS CloudTrail para solicitações feitas ao endpoint global do AWS STS são enviados para a região Leste dos EUA (Norte da Virgínia). Os logs do CloudTrail para solicitações atendidas por endpoints regionais do AWS STS continuarão sendo registrados em suas respectivas regiões no CloudTrail.
- Os logs do CloudTrail para operações realizadas pelo endpoint global do AWS STS e pelos endpoints regionais tem campos adicionais `endpointType` e `awsServingRegion` para indicar qual endpoint e região atenderam à solicitação. Para obter exemplos de log do CloudTrail, consulte Exemplo de evento de API AWS STS usando o endpoint global no arquivo de log do CloudTrail.
- As solicitações feitas ao endpoint global do AWS STS tem um valor de `us-east-1` para a chave de condição `aws:RequestedRegion`, independentemente de qual região atendeu à solicitação.
- As solicitações tratadas pelo endpoint global do AWS STS não compartilham uma cota de solicitações por segundo com endpoints regionais do AWS STS.

Se você tiver workloads em uma região opt-in e ainda estiver usando o endpoint global do AWS STS, recomendamos migrar para endpoints regionais do AWS STS para melhorar a resiliência e a performance. Para obter mais informações sobre a configuração de endpoints regionais do AWS

STS, consulte Endpoints regionais do AWS STS no Guia de referência de SDKs e ferramentas da AWS.

AWS CloudTrail e endpoints regionais

As chamadas para endpoints regionais e globais são registradas no campo `tlsDetails` no AWS CloudTrail. As chamadas para endpoints regionais, como `us-east-2.amazonaws.com`, são registradas no CloudTrail em sua região apropriada. As chamadas para o endpoint global, `sts.amazonaws.com`, são registradas como chamadas para um serviço global. Os eventos para endpoints globais de AWS STS são registrados em us-east-1.

> **Note**
>
> `tlsDetails` só pode ser visualizado para serviços que oferecem suporte a esse campo. Consulte Serviços que oferecem suporte a detalhes de TLS no CloudTrail no Guia de usuário do AWS CloudTrail
> Para obter mais informações, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.

Habilitar o acesso do intermediador de identidades personalizado ao console da AWS

Você pode escrever e executar um código para criar um URL que permita que os usuários que façam login na rede de sua organização acessem com segurança o Console de gerenciamento da AWS. O URL inclui um token de login que você obtém da AWS e que autentica o usuário na AWS. A sessão de console resultante pode incluir um `AccessKeyId` diferente devido à federação. Para rastrear o uso da chave de acesso para o login da federação por meio de eventos relacionados do CloudTrail, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail e Eventos de login do Console de gerenciamento da AWS.

> **Note**
>
> Se sua organização usa um provedor de identidade (IdP) que é compatível com SAML, é possível configurar o acesso ao console sem escrever código. Isso funciona com fornecedores como o Microsoft Active Directory Federation Services ou o Shibboleth de

código aberto. Para obter detalhes, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS.

Para permitir que os usuários de sua organização acessem o Console de gerenciamento da AWS, você pode criar um intermediador de identidades personalizado que executa as seguintes etapas:

1. Verificar se o usuário está autenticado pelo seu sistema de identidades local.
2. Chame as operações de API do AWS Security Token Service (AWS STS) AssumeRole (recomendado) ou GetFederationToken para obter credenciais de segurança temporárias para o usuário. Para saber mais sobre os diferentes métodos que você pode usar para assumir uma função, consulte Métodos para assumir um perfil. Para saber como passar tags de sessão opcionais ao obter suas credenciais de segurança, consulte Passar tags de sessão no AWS STS.
   - Se você usar uma das operações de API `AssumeRole*` para obter as credenciais de segurança temporárias para uma função, poderá incluir o parâmetro `DurationSeconds` na chamada. Esse parâmetro especifica a duração da sessão da função de 900 segundos (15 minutos) até o valor configurado de duração máxima da sessão para a função. Ao usar `DurationSeconds` em uma operação `AssumeRole*`, você deve chamá-la como um usuário do IAM com credenciais de longo prazo. Caso contrário, a chamada para o endpoint de federação na etapa 3 falhará. Para saber como visualizar ou alterar o valor máximo de uma função, consulte Atualizar a duração máxima da sessão de um perfil.
   - Se você usar a operação de API `GetFederationToken` para obter as credenciais, poderá incluir o parâmetro `DurationSeconds` na chamada. Esse parâmetro especifica a duração da sessão da função. O valor pode variar de 900 segundos (15 minutos) a 129.600 segundos (36 horas). Você só pode fazer essa chamada de API usando as credenciais de segurança da AWS de longo prazo de um usuário do IAM. Você também pode fazer essas chamadas usando credenciais de Usuário raiz da conta da AWS, mas isso não é recomendado. Se você fizer essa chamada como usuário raiz, a sessão padrão durará por uma hora. Ou você pode especificar uma sessão de 900 segundos (15 minutos) até 3.600 segundos (uma hora).
3. Chamar o endpoint de federação da AWS e fornecer as credenciais de segurança temporárias para solicitar um token de login.
4. Construir um URL para o console que inclui o token:
   - Se você usar uma das operações da API `AssumeRole*` em seu URL, poderá incluir o parâmetro HTTP `SessionDuration`. Esse parâmetro especifica a duração da sessão do console de 900 segundos (15 minutos) a 43.200 segundos (12 horas).

- Se você usar a operação de API `GetFederationToken` em seu URL, poderá incluir o parâmetro `DurationSeconds`. Esse parâmetro especifica a duração da sessão do console federado. O valor pode variar de 900 segundos (15 minutos) a 129.600 segundos (36 horas).

  > **Note**
  >
  > - A `SessionDuration` não pode ser maior ou igual à configuração de duração máxima da sessão para o perfil que você está assumindo. Por exemplo, você definiu a duração máxima da sessão para o perfil que deseja assumir como 5 horas. Seu parâmetro `SessionDuration` pode ser 16524 segundos ou 4 horas e 59 segundos.
  > - Não use o parâmetro HTTP `SessionDuration` quando obtiver as credenciais temporárias com `GetFederationToken`. A operação vai falhar.
  > - O uso de credenciais de uma função para assumir outra função é chamado de encadeamento de funções. Quando você usa o encadeamento de funções, suas novas credenciais são limitadas a uma duração máxima de uma hora. Quando você usa funções para conceder permissões a aplicativos executados em instâncias do EC2, esses aplicativos não estão sujeitos a essa limitação.
  > - Não use o parâmetro HTTP `SessionDuration` quando obtiver as credenciais temporárias por meio do encadeamento de perfis. A operação vai falhar.

5. Fornecer o URL para o usuário ou invocar o URL em nome do usuário.

O URL que o endpoint de federação fornece é válido por 15 minutos após sua criação. Esse valor é diferente da duração (em segundos) da sessão com credenciais de segurança temporárias que é associada ao URL. Essas credenciais são válidas pela duração especificada quando você as criou, a partir do momento em que elas foram criadas.

> **Important**
>
> O URL concede acesso aos seus recursos da AWS por meio do Console de gerenciamento da AWS se você habilitou permissões nas credenciais de segurança temporárias associadas. Por esse motivo, você deve tratar o URL como um segredo. Recomendamos o retorno do URL através de um redirecionamento seguro, por exemplo, usando um código de status de resposta HTTP 302 por meio de uma conexão SSL. Para obter mais informações sobre o código de status de resposta HTTP 302, consulte RFC 2616, seção 10.3.3.

Para concluir essas tarefas, você pode usar a API de consulta HTTPS para o AWS Identity and Access Management (IAM) e o AWS Security Token Service (AWS STS). Ou então, use linguagens de programação, como Java, Ruby ou C#, juntamente com o SDK da AWS. Cada um desses métodos é descrito nos tópicos a seguir.

Tópicos

- Código de exemplo usando operações de API de consulta do IAM
- Exemplo de código que usa o Python
- Exemplo de código usando Java
- Exemplo que mostra como criar o URL (Ruby)

Código de exemplo usando operações de API de consulta do IAM

Você pode construir um URL que conceda a perfis e entidades principais federadas acesso direto ao Console de gerenciamento da AWS. Esta tarefa usa a API de consulta HTTPS do IAM e do AWS STS. Para obter mais informações sobre como fazer solicitações de consulta, consulte Como fazer solicitações de consulta.

> **Note**
>
> O procedimento a seguir contém exemplos de strings de texto. Para melhorar a legibilidade, quebras de linha foram adicionadas em alguns dos exemplos mais longos. Quando você criar estas strings para seu próprio uso, você deve omitir as quebras de linha.

Para conceder a perfis e entidades principais federados acesso aos seus recursos via Console de gerenciamento da AWS

1. Autentique o usuário em seu sistema de identidades e autorização.
2. Obtenha credenciais de segurança temporárias para o usuário. As credenciais de segurança temporárias consistem em um ID de chave de acesso, uma chave de acesso secreta e um token de sessão. Para obter mais informações sobre a criação de credenciais de segurança temporárias, consulte Credenciais de segurança temporárias no IAM.

   Para obter credenciais temporárias, você pode chamar a API AssumeRole (recomendado) ou a API GetFederationToken do AWS STS. Para obter mais informações sobre as diferenças entre

essas operações de API, consulte Compreenda as opções de API para delegação segura de acesso à sua conta da AWS no blog de segurança da AWS.

> **Important**
>
> Quando você usa a API GetFederationToken para criar credenciais de segurança temporárias, deve especificar as permissões que as credenciais vão conceder ao usuário que assume a função. Para qualquer uma das operações de API que começam com `AssumeRole*`, você usa uma função do IAM para atribuir permissões. Para as outras operações de API, o mecanismo varia de acordo com a API. Para obter mais detalhes, consulte Permissões de credenciais de segurança temporárias. Além disso, se você usar as operações de API `AssumeRole*`, deverá chamá-las como um usuário do IAM com credenciais de longo prazo. Caso contrário, a chamada para o endpoint de federação na etapa 3 falhará.

3. Depois que você obter as credenciais de segurança temporárias, incorpore as credenciais em uma string de sessão JSON para trocá-las por um token de login. O exemplo a seguir mostra como codificar as credenciais. Substitua o espaço reservado para texto pelos valores apropriados das credenciais que você recebeu na etapa anterior.

```
{"sessionId":"*** temporary access key ID ***",
"sessionKey":"*** temporary secret access key ***",
"sessionToken":"*** session token ***"}
```

4. Codifique por URL a string da sessão da etapa anterior. Como as informações que você está codificando são confidenciais, recomendamos que você evite usar um serviço da web para esta codificação. Em vez disso, use uma função ou recurso instalado localmente em seu toolkit de desenvolvimento para codificar essas informações com segurança. Você pode usar a função `urllib.quote_plus` em Python, a função `URLEncoder.encode` em Java ou a função `CGI.escape` em Ruby. Veja os exemplos mais adiante neste tópico.

5.
> **Note**
>
> AWSA é compatível com solicitações POST aqui.

Envie a solicitação para o endpoint de federação da AWS:

`https://region-code.signin.aws.amazon.com/federation`

Para obter uma lista dos possíveis valores de *region-code* (região-código), consulte a coluna Region (Região) em AWS Sign-In endpoints (Endpoints de início de sessão da AWS). Opcionalmente, é possível usar um endpoint de federação de login padrão da AWS:

`https://signin.aws.amazon.com/federation`

A solicitação deve incluir os parâmetros `Action` e `Session`, e (opcionalmente) se você tiver usado uma operação de API `AssumeRole*`, um parâmetro HTTP `SessionDuration` conforme mostrado no exemplo a seguir.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> As instruções a seguir nesta etapa só funcionam usando solicitações GET.

O parâmetro HTTP `SessionDuration` especifica a duração da sessão do console federado. Ele é separado da duração das credenciais temporárias que você especifica usando o parâmetro `DurationSeconds`. Você pode especificar um valor `SessionDuration` máximo de 43200 (12 horas). Se o parâmetro `SessionDuration` estiver ausente, a sessão assume a duração das credenciais recuperadas do AWS STS na etapa 2 (cujo padrão é uma hora). Consulte a documentação da API `AssumeRole` para obter detalhes sobre como especificar a duração usando o parâmetro `DurationSeconds`. A capacidade de criar uma sessão do console com duração superior a uma hora é intrínseca para a operação `getSigninToken` do endpoint de federação.

> **Note**
>
> - A `SessionDuration` não pode ser maior ou igual à configuração de duração máxima da sessão para o perfil que você está assumindo. Por exemplo, você definiu a duração máxima da sessão para o perfil que deseja assumir como 5 horas. Seu parâmetro `SessionDuration` pode ser 16524 segundos ou 4 horas e 59 segundos.

- Não use o parâmetro HTTP `SessionDuration` quando obtiver as credenciais temporárias com `GetFederationToken`. A operação vai falhar.
- O uso de credenciais de uma função para assumir outra função é chamado de encadeamento de funções. Quando você usa o encadeamento de funções, suas novas credenciais são limitadas a uma duração máxima de uma hora. Quando você usa funções para conceder permissões a aplicativos executados em instâncias do EC2, esses aplicativos não estão sujeitos a essa limitação.
- Não use o parâmetro HTTP `SessionDuration` quando obtiver as credenciais temporárias por meio do encadeamento de perfis. A operação vai falhar.

Ao ativar as sessões do console com uma duração estendida, você aumenta o risco de exposição das credenciais. Para ajudar a reduzir esse risco, você pode desabilitar imediatamente as sessões ativas do console para qualquer função escolhendo a opção Revoke Sessions (Revogar sessões) em Role Summary (Resumo da função) na página do console do IAM. Para obter mais informações, consulte Revogar as credenciais de segurança temporárias do perfil do IAM.

Veja a seguir um exemplo da possível aparência de sua solicitação. As linhas são distribuídas aqui para legibilidade, mas você deve enviá-la como uma string de linha única.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

A resposta do endpoint de federação é um documento JSON com um valor `SigninToken`. Ele se parece com o seguinte exemplo.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWSA permite solicitações POST aqui.

Finalmente, crie o URL que seus usuários podem usar para acessar o Console de gerenciamento da AWS. O URL é o mesmo URL do endpoint de federação usado em Step 5, mais os seguintes parâmetros:

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> As instruções a seguir nesta etapa só funcionam usando a API GET.

O exemplo a seguir mostra a possível aparência do URL final. O URL é válido por 15 minutos a partir do momento em que ele é criada. As credenciais de segurança temporárias e a sessão do console incorporada dentro do URL são válidas durante o período especificado no parâmetro HTTP `SessionDuration` quando você inicialmente as solicita.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Exemplo de código que usa o Python

Os exemplos a seguir mostram como usar o Python para construir programaticamente um URL que ofereça aos usuários acesso direto ao Console de gerenciamento da AWS. Veja dois exemplos a seguir:

- Federar por meio de solicitações GET à AWS
- Federar por meio de solicitações POST à AWS

Ambos os exemplos usam o AWS SDK para Python (Boto3) e a API AssumeRole para obter credenciais de segurança temporárias.

Não inclua `SessionDuration` se as credenciais de `AssumeRoleSession` forem provenientes do encadeamento de perfis. Se você incluir `SessionDuration`, a operação vai falhar.

Usar solicitações GET

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your Conta da AWS,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
```

```
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)
```

Usar solicitações POST

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your AConta da AWS,
# call "AssumeRole" to get temporary access keys for the role or federated principal

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
```

```
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)
```

```
driver = webdriver.Chrome()
driver.execute_script(jsrequest)
input("Press Enter to close the browser window...")
```

Exemplo de código usando Java

O exemplo a seguir mostra como usar Java para construir programaticamente um URL que ofereça aos usuários acesso direto ao Console de gerenciamento da AWS. O trecho de código a seguir usa o AWS SDK for Java.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");
```

```
// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",       \"Statement\":[{\"Action\":\"sns:*
\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
                           URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);
```

```
// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Exemplo que mostra como criar o URL (Ruby)

O exemplo a seguir mostra como usar Ruby para construir programaticamente um URL que ofereça aos usuários acesso direto ao Console de gerenciamento da AWS. Esse trecho de código usa o AWS SDK for Ruby.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()
```

```
# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",       \"Statement\":{\"Effect\":\"Allow\",
\"Action\":\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
```

```
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Tags para recursos do AWS Identity and Access Management

Uma tag é um rótulo de atributo personalizado que você pode atribuir a um recurso da AWS. Cada tag tem duas partes:

- Uma chave de tag (por exemplo `CostCenter`, `Environment`, `Project` ou `Purpose`).
- Um campo opcional conhecido como um valor de tag (por exemplo, `111122223333`, `Production` ou um nome de equipe). Omitir o valor da tag é o mesmo que usar uma string vazia.

Juntos, esses são conhecidos como pares de chave-valor. Para saber os limites do número de etiquetas que você pode ter nos recursos do IAM, consulte IAM e cotas do AWS STS.

> **Note**
>
> Para obter detalhes sobre a distinção entre maiúsculas e minúsculas para chaves de tag e valores de chave de tag, consulte Case sensitivity.

As tags ajudam a identificar e organizar os recursos da AWS. Muitos serviços da AWS oferecem suporte à marcação para que você possa atribuir a mesma tag a recursos de diferentes serviços para indicar que os recursos estão relacionados. Por exemplo, você pode atribuir a mesma etiqueta a uma função do IAM que você atribui a um bucket do Amazon S3. Para obter mais informações sobre estratégias de marcação, consulte o Guia do usuário de Marcação de recursos da AWS.

Além de identificar, organizar e rastrear seus recursos do IAM com etiquetas, você pode usar etiquetas em políticas do IAM para ajudar a controlar quem pode visualizar e interagir com seus

recursos. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Tópicos

- Escolher uma convenção de nomenclatura de tag da AWS
- Regras para etiquetar no IAM e no AWS STS
- Marcar usuários do IAM
- Marcar perfis do IAM
- Marcar políticas gerenciadas pelo cliente
- Marcar provedores de identidades OpenID Connect (OIDC)
- Marcar provedores de identidades SAML do IAM
- Marcar perfis de instância para funções do Amazon EC2
- Marcar certificados do servidor
- Marcar dispositivos com MFA virtuais
- Marcar instâncias do gerenciador de acesso à conta
- Passar tags de sessão no AWS STS

Escolher uma convenção de nomenclatura de tag da AWS

Quando você começar a anexar etiquetas aos seus recursos do IAM, escolha sua convenção de nomenclatura de etiquetas com cuidado. Aplique a mesma convenção a todas as tags da AWS. Isso será especialmente importante se você usar tags em políticas para controlar acesso a recursos da AWS. Se você já usa tags em AWS, revise a convenção de nomenclatura e a ajuste de acordo.

> **Note**
>
> Se a conta for de um membro do AWS Organizations, consulte Políticas de tags no guia do usuário do AWS Organizations para saber mais sobre como usar tags no AWS Organizations.

Práticas recomendadas de nomenclatura de etiquetas

Estas são algumas práticas recomendadas e convenções de nomenclatura para etiquetas.

Certifique-se de que os nomes das tags sejam usados de forma consistente. Por exemplo, as tags `CostCenter` e `costcenter` são diferentes, então uma pode ser configurado como uma etiqueta de alocação de custos para análise financeira e relatórios e o outra pode não ser. Da mesma forma, a etiqueta `Name` aparece no AWS Console para muitos recursos, mas a etiqueta `name` não. Para obter detalhes sobre a distinção entre maiúsculas e minúsculas para chaves de tag e valores de chave de tag, consulte Case sensitivity.

Várias etiquetas são predefinidas pela AWS ou criadas automaticamente por diversos serviços da AWS. Muitos nomes de etiquetas definidos pela AWS usam todas as letras minúsculas, com hifens separando palavras no nome e prefixos para identificar o serviço de origem da etiqueta. Por exemplo:

- `aws:ec2spot:fleet-request-id` identifica a solicitação de instância spot do Amazon EC2 que iniciou a instância.
- `aws:cloudformation:stack-name` identifica a pilha da CloudFormation que criou o recurso.
- `elasticbeanstalk:environment-name` identifica a aplicação que criou o recurso.

Considere nomear suas etiquetas usando todas as letras minúsculas, com hifens separando palavras e um prefixo identificando o nome da organização ou o nome abreviado. Por exemplo, para uma empresa fictícia chamada AnyCompany, seria possível definir etiquetas como:

- `anycompany:cost-center` para identificar o código interno do centro de custos
- `anycompany:environment-type` para identificar se o ambiente é de desenvolvimento, teste ou produção
- `anycompany:application-id` para identificar a aplicação para a qual o recurso foi criado

O prefixo garante que as etiquetas sejam claramente identificadas como tendo sido definidas pela sua organização, e não por AWS nem por uma ferramenta de terceiros que você possa ter usado. Usar todas as letras minúsculas com hifens para separadores evita confusão sobre como formatar o nome de uma etiqueta em letras maiúsculas. Por exemplo, `anycompany:project-id` é mais simples de lembrar do que `ANYCOMPANY:ProjectID`, `anycompany:projectID` ou `Anycompany:ProjectId`.

Regras para etiquetar no IAM e no AWS STS

Uma série de convenções regem a criação e a aplicação de etiquetas no IAM e no AWS STS.

Nomear tags

Observe as seguintes convenções ao formular uma convenção de nomenclatura de etiqueta para recursos do IAM, sessões de assumir função do AWS STS e sessões de usuário federado do AWS STS:

Requisitos de caracteres: as chaves e os valores de etiqueta podem incluir qualquer combinação de letras, números, espaços e dos símbolos _ . : / = + - @.

Diferenciação entre maiúsculas e minúsculas: a diferenciação de maiúsculas e minúsculas para chaves de etiqueta difere de acordo com o tipo de recurso do IAM que é etiquetado. Os valores de chave de etiqueta para usuários e funções do IAM não diferenciam maiúsculas de minúsculas, mas são preservados. Isso significa que você não pode ter chaves de tag **`Department`** e **`department`** separadas. Se você tiver marcado um usuário com a tag **`Department=finance`** e adicionar a tag **`department=hr`**, ela substituirá a primeira. Uma segunda tag não é adicionada.

Para outros tipos de recursos do IAM, os valores de chave de etiqueta diferenciam maiúsculas de minúsculas. Isso significa que você pode ter as chaves de tag **`Costcenter`** e **`costcenter`**. Por exemplo, se você tiver marcado uma política gerenciada pelo cliente com a tag **`Costcenter = 1234`** e adicionar a tag **`costcenter = 5678`**, a política terá ambas as chaves de tag **`Costcenter`** e **`costcenter`**.

Como prática recomendada, recomendamos que você evite usar tags semelhantes com padrões diferentes de maiúsculas e minúsculas. Recomendamos definir uma estratégia para letras maiúsculas em etiquetas e implementá-las de forma consistente em todos os tipos de recursos. Para saber mais sobre as melhores práticas para marcação, consulte Marcar recursos da AWS na Referência geral da AWS.

As listas a seguir mostram as diferenças na distinção entre maiúsculas e minúsculas para chaves de etiqueta que são anexadas aos recursos do IAM.

Os valores de chave de tag não diferenciam maiúsculas de minúsculas:

- Perfis do IAM
- Usuários do IAM

Os valores de chave de tag diferenciam maiúsculas de minúsculas:

- Políticas gerenciadas pelo cliente
- Perfis de instância
- Provedores de identidade do OpenID Connect
- Provedores de identidade SAML
- Certificados de servidor
- Dispositivos MFA virtuais

Além disso, as seguintes regras se aplicam:

- Você não pode criar uma chave ou um valor de tag que comece com o texto **`aws:`**. Esse prefixo de etiqueta está reservado para uso interno da AWS.
- Você pode criar uma tag com um valor vazio, como **`phoneNumber =`** . Você não pode criar uma chave de tag vazia.
- Você não pode especificar vários valores em uma única tag, mas pode criar uma estrutura multivalor personalizada no valor único. Por exemplo, suponhamos que o usuário Zhang trabalhe na equipe de engenharia e na equipe de controle de qualidade. Se anexar a tag **`team = Engineering`** e, em seguida, anexar a tag **`team = QA`**, você alterará o valor da tag de **`Engineering`** para **`QA`**. Em vez disso, você pode incluir vários valores em uma única tag com um separador personalizado. Neste exemplo, você pode anexar a tag **`team = Engineering:QA`** a Zhang.

> **Note**
>
> Para controlar o acesso a engenheiros neste exemplo usando a tag **`team`**, você deve criar uma política que possibilite todas as configurações que possam incluir **`Engineering`**, até mesmo **`Engineering:QA`**. Para saber mais sobre como usar tags em políticas, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Aplicar e editar tags

Observe as seguintes convenções ao associar etiquetas a recursos do IAM:

- Você pode marcar a maioria dos recursos do IAM, mas não grupos, funções assumidas, relatórios de acesso e dispositivos de MFA baseada em hardware.
- Você não pode usar o Tag Editor para etiquetar recursos do IAM. O Tag Editor não é compatível com etiquetas do IAM. Para obter informações sobre como usar o Tag Editor com outros serviços, consulte Working with Tag Editor no Guia do usuário do Grupos de recursos da AWS.
- Para etiquetar um recurso do IAM, você deve ter permissões específicas. Para marcar ou desmarcar recursos, você também deve ter permissão para listar tags. Para obter mais informações, consulte a lista de tópicos para cada recurso do IAM no final desta página.
- O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.
- Você pode aplicar a mesma etiqueta a vários recursos do IAM. Por exemplo, suponha que você tenha um departamento chamado `AWS_Development` com 12 membros. É possível ter 12 usuários e uma função com a chave de tag de **`department`** e um valor de **`awsDevelopment`** (**`department = awsDevelopment`**). Você também pode usar a mesma tag em recursos em outros serviços que dão suporte a tags.
- Entidades (usuários ou funções) do IAM não podem ter várias instâncias da mesma chave de etiqueta. Por exemplo, se você tiver um usuário com o par de chave-valor de etiqueta **`costCenter = 1234`**, poderá associar o par de chave-valor da etiqueta **`costCenter = 5678`**. O IAM atualiza o valor da etiqueta **`costCenter`** para **`5678`**.
- Para editar uma etiqueta associada a uma entidade (usuário ou função) do IAM, associe uma etiqueta com um novo valor para substituir a etiqueta existente. Por exemplo, suponhamos que você tenha um usuário com o par de chave-valor de tag **`department = Engineering`**. Se precisar mover o usuário para o departamento de controle de qualidade, você poderá associar o par de chave-valor de tag **`department = QA`** a ele. Isso resulta na substituição do **`Engineering`** valor da chave de tag **`department`** pelo valor **`QA`**.

Marcar usuários do IAM

Você pode usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados a um usuário do IAM. Por exemplo, para adicionar informações de localização a um usuário, você pode adicionar a chave de tag **`location`** e o valor de tag **`us_wa_seattle`**. Ou você pode usar três pares de chave-valor de tags de locais separados: **`loc-country = us`**, **`loc-state = wa`** e **`loc-city = seattle`**. Você pode usar tags para controlar o acesso de um usuário a recursos ou controlar quais tags podem ser associadas a um usuário. Para saber mais sobre como usar

tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Permissões necessárias para etiquetar usuários do IAM

Você deve configurar permissões para permitir que um usuário do IAM possa etiquetar outros usuários. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Permitir que um usuário do IAM adicione, liste ou remova uma etiqueta para um usuário específico

Adicione a instrução a seguir à política de permissões do usuário do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *<username>* pelo nome do usuário cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Para permitir que um usuário do IAM autogerencie etiquetas

Adicione a seguinte instrução à política de permissões para que usuários permitam que outros gerenciem as próprias tags. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

Para permitir que um usuário do IAM adicione uma etiqueta a um usuário específico

Adicione a seguinte instrução à política de permissões para o usuário do IAM que precisa adicionar, mas não remover, etiquetas para um usuário específico.

> **Note**
>
> A ação `iam:TagUser` requer que você também inclua a ação `iam:ListUserTags`.

Para usar essa política, substitua *<username>* pelo nome do usuário cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciamento de etiquetas em usuários do IAM (console)

Você pode gerenciar etiquetas de usuários do IAM no Console de gerenciamento da AWS.

Gerenciar tags em usuários (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Users (usuários) e, em seguida, escolha o nome do usuário que deseja editar.
3. Escolha a guia Tags e conclua uma das seguintes ações:
   - Escolha Adicionar nova etiqueta se o usuário ainda não tiver etiquetas.
   - Escolha Manage tags (Gerenciar tags) para gerenciar o conjunto de tags existente.
4. Adicione ou remova tags para concluir o conjunto de tags. Em seguida, escolha Salvar alterações.

Gerenciamento de etiquetas em usuários do IAM (AWS CLI ou API da AWS)

Você pode listar, anexar ou remover etiquetas de usuários do IAM. Você pode usar a AWS CLI ou a API da AWS para gerenciar etiquetas de usuários do IAM.

Para listar as etiquetas atualmente anexadas a um usuário do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-user-tags
- AWS API: ListUserTags

Para anexar etiquetas a um usuário do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam tag-user
- AWS API: TagUser

Para remover etiquetas de um usuário do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam untag-user
- AWS API: UntagUser

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar perfis do IAM

Você pode usar pares de chave-valor de etiquetas do IAM para adicionar atributos personalizados a uma função do IAM. Por exemplo, para adicionar informações de localização a uma função, você pode adicionar a chave de tag **`location`** e o valor de tag **`us_wa_seattle`**. Ou você pode usar três pares de chave-valor de tags de locais separados: **`loc-country = us`**, **`loc-state = wa`** e **`loc-city = seattle`**. Você pode usar tags para controlar o acesso de uma função a recursos ou controlar quais tags podem ser associadas a uma função. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Permissões necessárias para etiquetar funções do IAM

Você deve configurar permissões para permitir que uma função do IAM etiquete outras entidades (usuários ou funções). Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListRoleTags`
- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Para permitir que uma função do IAM adicione, liste ou remova uma etiqueta para um usuário específico

Adicione a instrução a seguir à política de permissões para a função do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *`<username>`* pelo nome do usuário cujas tags

precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Para permitir que uma função do IAM adicione uma etiqueta para um usuário específico

Adicione a seguinte instrução à política de permissões para a função do IAM que precisa adicionar, mas não remover, etiquetas para um usuário específico.

Para usar essa política, substitua *<username>* pelo nome do usuário cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Para permitir que uma função do IAM adicione, liste ou remova uma etiqueta para uma função específica

Adicione a instrução a seguir à política de permissões para a função do IAM que precisa gerenciar etiquetas. Substitua *<rolename>* pelo nome da função cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciamento de etiquetas em funções do IAM (console)

Você pode gerenciar etiquetas de funções do IAM no Console de gerenciamento da AWS.

Gerenciar tags em funções (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Roles (Funções) e, em seguida, escolha o nome da função que deseja editar.
3. Escolha a guia Tags e conclua uma das seguintes ações:
   - Escolha Add tags (Adicionar tags) se a função ainda não tiver tags.
   - Escolha Manage tags (Gerenciar tags) para gerenciar o conjunto de tags existente.
4. Adicione ou remova tags para concluir o conjunto de tags. Depois, escolha Save changes (Salvar alterações).

Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS)

Você pode listar, anexar ou remover etiquetas de funções do IAM. Você pode usar a AWS CLI ou a API da AWS para gerenciar etiquetas em funções do IAM.

Para listar as etiquetas atualmente anexadas a uma função do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-role-tags
- API da AWS: ListRoleTags

Para anexar etiquetas a uma função do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam tag-role
- AWS API: TagRole

Para remover etiquetas de uma função do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam untag-role
- AWS API: UntagRole

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar políticas gerenciadas pelo cliente

Você pode usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados às políticas gerenciadas pelo cliente. Por exemplo, para marcar uma política com informações de departamento, você pode adicionar a chave de tag **`Department`** e o valor de tag **`eng`**. Ou, talvez você queira marcar políticas para indicar que elas se aplicam a um ambiente específico, como **`Environment = lab`**. Você pode usar tags para controlar o acesso a recursos ou controlar quais tags podem ser associadas a um recurso. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Permissões necessárias para marcar políticas gerenciadas pelo cliente

Você deve configurar permissões para permitir que uma entidade (usuário ou função) do IAM possa etiquetar políticas gerenciadas pelo cliente. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListPolicyTags`
- `iam:TagPolicy`

- `iam:UntagPolicy`

Para permitir que uma entidade do IAM (usuário ou função) adicione, liste ou remova uma etiqueta para uma política gerenciada pelo cliente

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *<policyname>* pelo nome da política cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Para permitir que uma entidade do IAM (usuário ou função) adicione uma etiqueta a uma política específica gerenciada pelo cliente

Adicione a instrução a seguir à política de permissões para a entidade do IAM que precisa adicionar, mas não remover, etiquetas para uma política específica.

> **Note**
>
> A ação `iam:TagPolicy` requer que você também inclua a ação `iam:ListPolicyTags`.

Para usar essa política, substitua *<policyname>* pelo nome da política cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
```

```
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciamento de etiquetas em políticas gerenciadas pelo cliente do IAM (console)

Você pode gerenciar etiquetas para políticas gerenciadas pelo cliente do IAM do Console de gerenciamento da AWS.

Gerenciar tags em políticas gerenciadas pelo cliente (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Policies (Políticas) e, em seguida, escolha o nome da política gerenciada pelo cliente que deseja editar.
3. Escolha a guia Tags e depois escolha Gerenciar tags.
4. Adicione ou remova tags para concluir o conjunto de tags. Em seguida, escolha Salvar alterações.

Gerenciamento de etiquetas em políticas gerenciadas pelo cliente do IAM (AWS CLI ou API da AWS)

Você pode listar, anexar ou remover etiquetas para políticas gerenciadas pelo cliente IAM. Você pode usar a AWS CLI ou a API da AWS para gerenciar etiquetas para políticas gerenciadas pelo cliente do IAM.

Para listar as etiquetas atualmente anexadas a uma política gerenciada pelo cliente do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-policy-tags
- AWS API: ListPolicyTags

Para anexar etiquetas a uma política gerenciada pelo cliente do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam tag-policy

- AWS API: TagPolicy

Para remover etiquetas de uma política gerenciada pelo cliente do IAM (AWS CLI ou API da AWS)

- AWS CLI: aws iam untag-policy
- AWS API: UntagPolicy

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar provedores de identidades OpenID Connect (OIDC)

Você pode usar chaves-valores de etiqueta do IAM para adicionar atributos personalizados a provedores de identidade OpenID Connect (OIDC) do IAM. Por exemplo, para identificar um provedor de identidade OIDC, você pode adicionar a chave de tag **`google`** e o valor de tag **`oidc`**. Você pode usar tags para controlar o acesso a recursos ou controlar quais tags podem ser associadas a um objeto. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Permissões necessárias para etiquetar provedores de identidade OIDC do IAM

Você deve configurar permissões para permitir que uma entidade (usuário ou função) do IAM etiquete provedores de identidade OIDC do IAM. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

Para permitir que uma entidade do IAM adicione, liste ou remova uma etiqueta para um provedor de identidades OIDC do IAM

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *`<OIDCProviderName>`* pelo nome do provedor de identidade OIDC cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma

política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Para permitir que uma entidade (usuário ou função) do IAM adicione uma etiqueta a um provedor de identidade OIDC específico do IAM

Adicione a instrução a seguir à política de permissões da entidade do IAM que precisa adicionar, mas não remover, etiquetas em um provedor de identidade específico.

> **Note**
>
> A ação `iam:TagOpenIDConnectProvider` requer que você também inclua a ação `iam:ListOpenIDConnectProviderTags`.

Para usar essa política, substitua *<OIDCProviderName>* pelo nome do provedor de identidade OIDC cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciamento de etiquetas em provedores de identidade OIDC do IAM (console)

Você pode gerenciar etiquetas para provedores de identidade OIDC do IAM no Console de gerenciamento da AWS.

Gerenciar tags em provedores de identidade OIDC (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Identity providers (Provedores de identidade) e, em seguida, escolha o nome do provedor de identidade que deseja editar.
3. Escolha a guia Tags e, na seção Tags, selecione Gerenciar tags e, em seguida, conclua uma das seguintes ações:
   - Escolha Add tag (Adicionar tag) se o provedor de identidade OIDC ainda não tiver tags ou para adicionar uma nova tag.
   - Edite chaves e valores de tag existentes.
   - Para remover uma tag, escolha Remove tag (Remover tag).
4. Em seguida, escolha Salvar alterações.

Gerenciamento de etiquetas em provedores de identidade OIDC do IAM (AWS CLI ou API da AWS)

Você pode listar, anexar ou remover etiquetas em provedores de identidade OIDC do IAM. Você pode usar a AWS CLI ou a API da AWS para gerenciar etiquetas em provedores de identidade OIDC do IAM.

Para listar as etiquetas atualmente anexadas a um provedor de identidade OIDC do IAM (AWS CLI ou AWS API)

- AWS CLI: aws iam list-open-id-connect-provider-tags
- AWS API: ListOpenIDConnectProviderTags

Para anexar etiquetas a um provedor de identidade OIDC do IAM (AWS CLI ou AWS API)

- AWS CLI: aws iam tag-open-id-connect-provider
- AWS API: TagOpenIDConnectProvider

Para remover etiquetas de um provedor de identidade OIDC do IAM (AWS CLI ou AWS API)

- AWS CLI: aws iam untag-open-id-connect-provider
- AWS API: UntagOpenIDConnectProvider

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar provedores de identidades SAML do IAM

Você pode usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados a provedores de identidade SAML. Por exemplo, para identificar um provedor, você pode adicionar a chave de tag **`okta`** e o valor de tag **`saml`**. Você pode usar tags para controlar o acesso a recursos ou controlar quais tags podem ser associadas a um objeto. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Permissões necessárias para marcar provedores de identidade SAML

Você deve configurar permissões para permitir que uma entidade (usuários ou funções) do IAM etiquete provedores de identidade (IdPs) baseados no SAML 2.0. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

Para permitir que uma entidade (usuário ou função) do IAM adicione, liste ou remova uma etiqueta em um provedor de identidade SAML

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *`<SAMLProviderName>`* pelo nome do provedor SAML cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Para permitir que uma entidade (usuário ou função) do IAM adicione uma etiqueta a um provedor de identidade SAML específico

Adicione a instrução a seguir à política de permissões da entidade do IAM que precisa adicionar, mas não remover, etiquetas em um provedor SAML específico.

> **Note**
>
> A ação `iam:TagSAMLProvider` requer que você também inclua a ação `iam:ListSAMLProviderTags`.

Para usar essa política, substitua *<SAMLProviderName>* pelo nome do provedor SAML cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciamento de etiquetas em provedores de identidade SAML do IAM (console)

Você pode gerenciar etiquetas para provedores de identidade SAML do IAM no Console de gerenciamento da AWS.

Gerenciar tags em provedores de identidade SAML (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Identity providers (Provedores de identidade) e, em seguida, escolha o nome do provedor de identidade SAML que deseja editar.
3. Escolha a guia Tags e, na seção Tags, selecione Gerenciar tags e, em seguida, conclua uma das seguintes ações:
   - Escolha Add tag (Adicionar tag) se o provedor de identidade SAML ainda não tiver tags ou para adicionar uma nova tag.
   - Edite chaves e valores de tag existentes.
   - Para remover uma tag, escolha Remove tag (Remover tag).
4. Adicione ou remova tags para concluir o conjunto de tags. Em seguida, escolha Salvar alterações.

Gerenciamento de etiquetas em provedores de identidade SAML do IAM (AWS CLI ou API da AWS)

Você pode listar, anexar ou remover etiquetas em provedores de identidade SAML do IAM. Você pode usar a AWS CLI ou a API da AWS para gerenciar etiquetas em provedores de identidade SAML do IAM.

Listar tags atualmente associadas a um provedor de identidade SAML (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-saml-provider-tags
- AWS API: ListSAMLProviderTags

Associar tags a um provedor de identidade SAML (AWS CLI ou API da AWS)

- AWS CLI: aws iam tag-saml-provider
- AWS API: TagSAMLProvider

Remover tags de um provedor de identidade SAML (AWS CLI ou API da AWS)

- AWS CLI: aws iam untag-saml-provider
- AWS API: UntagSAMLProvider

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar perfis de instância para funções do Amazon EC2

Quando você executa uma instância do Amazon EC2, você especifica uma função do IAM para associar à instância. Perfil de instância é um contêiner para uma função do IAM que pode ser usada para passar informações da função para uma instância do Amazon EC2 quando a instância é iniciada. Você pode marcar perfis de instância ao usar a AWS CLI ou a API da AWS.

É possível usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados a um perfil de instância. Por exemplo, para adicionar informações de departamento a um perfil de instância, você pode adicionar a chave de tag **`access-team`** e o valor de tag **`eng`**. Assim, os principais com tags correspondentes terão acesso a perfis de instância com a mesma tag. Você pode usar vários pares de chave-valor de tag para especificar uma equipe e um projeto: **`access-team = eng`** e **`project = peg`**. Você pode usar tags para controlar o acesso de um usuário a recursos ou controlar quais tags podem ser associadas a um usuário. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Permissões necessárias para marcar perfis de instância

Você deve configurar permissões para permitir que uma entidade (usuário ou função) do IAM etiquete perfis de instância. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListInstanceProfileTags`

- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

Para permitir que uma entidade (usuário ou função) do IAM adicione, liste ou remova uma etiqueta em um perfil de instância

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *<InstanceProfileName>* pelo nome do perfil de instância cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Para permitir que uma entidade (usuário ou função) do IAM adicione uma etiqueta a um perfil de instância específico

Adicione a declaração a seguir à política de permissões da entidade do IAM que precisa adicionar, mas não remover, etiquetas em um perfil de instância específico.

> **Note**
>
> A ação `iam:TagInstanceProfile` requer que você também inclua a ação `iam:ListInstanceProfileTags`.

Para usar essa política, substitua *<InstanceProfileName>* pelo nome do perfil de instância cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciar tags em perfis de instância (AWS CLI ou API da AWS)

Você pode listar, associar ou remover tags em perfis de instância. Você pode usar a AWS CLI ou a API da AWS para gerenciar tags em perfis de instância.

Listar tags atualmente associadas a um perfil de instância (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-instance-profile-tags
- AWS API: ListInstanceProfileTags

Associar tags a um perfil de instância (AWS CLI ou AWS API)

- AWS CLI: aws iam tag-instance-profile
- AWS API: TagInstanceProfile

Remover tags de um perfil de instância (AWS CLI ou AWS API)

- AWS CLI: aws iam untag-instance-profile
- AWS API: UntagInstanceProfile

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar certificados do servidor

Se você usar o IAM para gerenciar certificados SSL/TLS, poderá etiquetar os certificados do servidor no IAM usando AWS CLI ou a API da AWS. Para certificados em uma região compatível com Gerenciador de certificados da AWS (ACM), recomendamos que você use o ACM em vez do IAM para provisionar, gerenciar e implantar seus certificados de servidor. Nas regiões sem suporte, você deve usar o IAM como gerenciador de certificados. Para saber quais regiões são compatíveis com o ACM, consulte Cotas e endpoints do Gerenciador de certificados da AWS na Referência geral da AWS.

Você pode usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados a um certificado de servidor. Por exemplo, para adicionar informações sobre o proprietário ou administrador de um certificado de servidor, adicione a chave de tag **`owner`** e o valor de tag **`net-eng`**. Ou você pode especificar um centro de custo adicionando a chave de tag **`CostCenter`** e o valor de tag **`1234`**. Você pode usar tags para controlar o acesso a recursos ou controlar quais tags podem ser associadas a um recurso. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Permissões necessárias para marcar certificados de servidor

Você deve configurar permissões para permitir que uma entidade (usuário ou função) do IAM etiquete certificados de servidor. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

Para permitir que uma entidade (usuário ou função) do IAM adicione, liste ou remova uma etiqueta em um certificado de servidor

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *`<CertificateName>`* pelo nome do certificado de servidor cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política

usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Para permitir que uma entidade (usuário ou função) do IAM adicione uma etiqueta a um certificado de servidor específico

Adicione a instrução a seguir à política de permissões da entidade do IAM que precisa adicionar, mas não remover, etiquetas em um certificado de servidor específico.

> **Note**
>
> A ação `iam:TagServerCertificate` requer que você também inclua a ação `iam:ListServerCertificateTags`.

Para usar essa política, substitua *<CertificateName>* pelo nome do certificado de servidor cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciar tags em certificados de servidor (AWS CLI ou API da AWS)

Você pode listar, associar ou remover tags em certificados de servidor. Você pode usar a AWS CLI ou a API da AWS para gerenciar tags em certificados de servidor.

Listar tags atualmente associadas a um certificado de servidor (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-server-certificate-tags
- AWS API: ListServerCertificateTags

Associar tags a um certificado de servidor (AWS CLI ou API da AWS)

- AWS CLI: aws iam tag-server-certificate
- AWS API: TagServerCertificate

Remover tags de um certificado de servidor (AWS CLI ou API da AWS)

- AWS CLI: aws iam untag-server-certificate
- AWS API: UntagServerCertificate

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar dispositivos com MFA virtuais

Você pode usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados a um dispositivo com MFA virtual. Por exemplo, para adicionar informações do centro de custo para o dispositivo MFA virtual de um usuário, você pode adicionar a chave de tag **`CostCenter`** e o valor de tag **`1234`**. Você pode usar tags para controlar o acesso a recursos ou controlar quais tags podem ser associadas a um objeto. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para obter mais informações, consulte Passar tags de sessão no AWS STS.

Permissões necessárias para marcar dispositivos MFA virtuais

Você deve configurar permissões para permitir que uma entidade (usuário ou função) do IAM etiquete dispositivos com MFA virtuais. Você pode especificar uma ou todas as seguintes de etiqueta do IAM em uma política do IAM:

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`
- `iam:UntagMFADevice`

Para permitir que uma entidade (usuário ou função) do IAM adicione, liste ou remova uma etiqueta em um dispositivo com MFA virtual

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da sua conta e substitua *<MFATokenID>* pelo nome do dispositivo MFA virtual cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Para permitir que uma entidade (usuário ou função) do IAM adicione uma etiqueta a um dispositivo com MFA virtual específico

Adicione a seguinte instrução à política de permissões da entidade do IAM que precisa adicionar, mas não remover, etiquetas em um dispositivo com MFA específico.

> **Note**
>
> A ação `iam:TagMFADevice` requer que você também inclua a ação `iam:ListMFADeviceTags`.

Para usar essa política, substitua *<MFATokenID>* pelo nome do dispositivo MFA virtual cujas tags precisam ser gerenciadas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Como alternativa, você pode usar uma política gerenciada pela AWS, como IAMFullAccess, para fornecer acesso total ao IAM.

Gerenciar tags em dispositivos MFA virtuais (AWS CLI ou API da AWS)

Você pode listar, associar ou remover tags em um dispositivo MFA virtual. Você pode usar a AWS CLI ou a API da AWS para gerenciar tags em um dispositivo MFA virtual.

Listar tags atualmente associadas a um dispositivo MFA virtual (AWS CLI ou API da AWS)

- AWS CLI: aws iam list-mfa-device-tags
- AWS API: ListMFADeviceTags

Associar tags a um dispositivo MFA virtual (AWS CLI ou API da AWS)

- AWS CLI: aws iam tag-mfa-device
- AWS API: TagMFADevice

Remover tags de um dispositivo MFA virtual (AWS CLI ou API da AWS)

- AWS CLI: aws iam untag-mfa-device
- AWS API: UntagMFADevice

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte Elementos de política do IAM: variáveis e etiquetas.

Marcar instâncias do gerenciador de acesso à conta

Você deve configurar as permissões para permitir que uma entidade do IAM (perfis ou usuários) marque instâncias do gerenciador de acesso à conta. Você pode especificar uma ou todas as seguintes ações de tag de acesso à conta em uma política do IAM:

- `account-access:TagResource`
- `account-access:ListTagsForResource`
- `account-access:UntagResource`

Para permitir que uma entidade (perfil ou usuário) do IAM adicione, liste ou remova uma tag em uma instância do gerenciador de acesso à conta

Adicione a seguinte instrução à política de permissões para a entidade do IAM que precisa gerenciar etiquetas. Use o número da conta e os IDs de instância do IAM Identity Center e das instâncias do gerenciador de acesso à conta para substituir os espaços reservados. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas do IAM (console)”.

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource",
        "account-access:UntagResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

Para permitir que uma entidade (perfil ou usuário) do IAM adicione uma tag a uma instância específica do gerenciador de acesso à conta

Adicione a instrução a seguir à política de permissões para a entidade do IAM que precisa adicionar, mas não remover, tags para uma instância específica do gerenciador de acesso à conta.

> **Note**
>
> A ação `account-access:TagResource` requer que você também inclua a ação `account-access:ListTagsForResource`.

Use o número da conta e os IDs de instância do IAM Identity Center e das instâncias do gerenciador de acesso à conta para substituir os espaços reservados. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas do IAM (console)”.

```
{
    "Effect": "Allow",
    "Action": [
        "account-access:TagResource",
        "account-access:ListTagsForResource"
    ],
    "Resource": "<account_access_manager_ARN>"
}
```

Gerenciar tags no gerenciador de acesso à conta (console)

Você pode gerenciar tags para uma instância do gerenciador de acesso à conta do Console de gerenciamento da AWS.

Gerenciar tags no gerenciador de acesso à conta (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console, escolha Gerenciador de acesso à conta e, em seguida, escolha a guia Configurações.
3. Na seção Tags, você pode ver as tags atualmente configuradas. Escolha Gerenciar tags e conclua uma das seguintes ações:
   - Selecione Adicionar nova tag para adicionar uma tag.
   - Edite chaves e valores de tag existentes.
   - Selecione Remover para remover uma tag.

4. Adicione ou remova tags para concluir o conjunto de tags. Em seguida, escolha Salvar alterações.

Gerenciamento de tags no gerenciador de acesso à conta (AWS CLI ou API da AWS)

Você pode listar, associar ou remover tags em instâncias do gerenciador de acesso à conta usando a AWS CLI ou a API da AWS.

Para listar as tags atualmente associadas a uma instância do gerenciador de acesso à conta (AWS CLI ou API da AWS)

- AWS CLI: `aws account-access list-tags-for-resource`
- AWS API: `account-access:ListTagsForResource`

Para anexar tags a uma instância do gerenciador de acesso à conta (AWS CLI ou API da AWS)

- AWS CLI: `aws account-access tag-resource`
- AWS API: `TagResource`

Para remover tags de uma instância do gerenciador de acesso à conta (AWS CLI ou API da AWS)

- AWS CLI: `aws account-access untag-resource`
- AWS API: `UntagResource`

Para obter informações sobre como anexar tags a recursos de outros serviços da AWS, consulte a documentação desses serviços.

Para obter informações sobre como usar etiquetas para definir mais permissões detalhadas com políticas de permissões do IAM, consulte the section called “Variáveis e tags”.

Passar tags de sessão no AWS STS

As etiquetas de sessão são atributos de par de chave-valor que você passa ao assumir uma função do IAM ou federar um usuário no AWS STS. Isso é feito criando uma solicitação da AWS CLI ou da API da AWS por meio do AWS STS ou do seu provedor de identidade (IdP). Ao usar o AWS STS para solicitar credenciais de segurança temporárias, você gera uma sessão. As sessões expiram e têm credenciais, tais como um par de chaves de acesso e um token de sessão. Quando você usa

as credenciais de sessão para fazer uma solicitação subsequente, o contexto de solicitação inclui a chave de contexto `aws:PrincipalTag`. Você pode usar a chave `aws:PrincipalTag` no elemento `Condition` de suas políticas para permitir ou negar acesso com base nessas tags.

Ao usar credenciais temporárias para fazer uma solicitação, seu principal pode incluir um conjunto de tags. Essas tags vêm das seguintes fontes:

1. Tags de sessão: as etiquetas que foram passadas quando você assumiu a função ou federou o usuário usando a AWS CLI ou a API da AWS. Para obter mais informações sobre essas operações, consulte a Operações de marcação de sessão.
2. Etiquetas de sessão transitivas recebidas: essas etiquetas foram herdadas de uma sessão anterior em uma cadeia de funções. Para obter mais informações, consulte Encadeamento de funções com tags de sessão mais adiante neste tópico.
3. Etiquetas do IAM: as etiquetas anexadas à sua função assumida no IAM.

Tópicos

- Operações de marcação de sessão
- Coisas a saber sobre tags de sessão
- Permissões necessárias para adicionar tags de sessão
- Passar tags de sessão usando AssumeRole
- Passar tags de sessão usando AssumeRoleWithSAML
- Passar tags de sessão usando AssumeRoleWithWebIdentity
- Passar tags de sessão usando GetFederationToken
- Encadeamento de funções com tags de sessão
- Usar tags de sessão para ABAC
- Visualizar etiquetas da sessão no CloudTrail

Operações de marcação de sessão

Você pode passar tags de sessão usando as seguintes operações do AWS CLI ou API da AWS em AWS STS. O recurso Switch Role (Alternar função) do Console de gerenciamento da AWS não permite que você passe tags de sessão.

Você também pode definir as tags de sessão como transitivas. As tags transitivas persistem durante o encadeamento de funções. Para obter mais informações, consulte Encadeamento de funções com tags de sessão.

A tabela a seguir compara os métodos de passagem de tags de sessão.

Operation	Quem pode assumir a função	Método para passar tags	Método para definir tags transitivas
Operação da CLI `assume-role` ou da API `AssumeRole`	Usuário ou sessão do IAM	Parâmetro da API `Tags` ou opção da CLI `--tags`	Parâmetro da API `TransitiveTagKeys` ou opção da CLI `--transitive-tag-keys`
Operação da CLI `assume-role-with-saml` ou da API `AssumeRoleWithSAML`	Qualquer usuário autenticado usando um provedor de identidade SAML	`PrincipalTag` Atributo SAML	`TransitiveTagKeys` Atributo SAML
Operação da CLI `assume-role-with-web-identity` ou da API `AssumeRoleWithWebIdentity`	Qualquer usuário autenticado usando um provedor OIDC	Token OIDC `PrincipalTag`	Token OIDC `TransitiveTagKeys`
Operação da CLI `get-federation-token` ou da API `GetFederationToken`	Usuário raiz ou usuário do IAM	Parâmetro da API `Tags` ou opção da CLI `--tags`	Não compatível

As operações que oferecem suporte à etiquetamento de sessão podem falhar nas seguintes condições:

- Você passar mais de 50 tags de sessão.
- O texto simples das chaves de etiqueta de sessão exceder 128 caracteres.
- O texto simples dos valores de etiqueta de sessão exceder 256 caracteres.
- O tamanho total do texto simples das políticas da sessão exceder 2048 caracteres.
- As políticas de sessão e as tags de sessão aumentam o tamanho do token da sessão. Se o token exceder o limite de tamanho, a operação falhará. A mensagem de erro informa o tamanho do token da sessão e o tamanho máximo permitido, ambos em bytes. Monitore `SessionTokenUtilization` na resposta da API AWS STS para rastrear seu uso.

Coisas a saber sobre tags de sessão

Antes de usar tags de sessão, revise os seguintes detalhes sobre sessões e tags.

- Ao usar etiquetas de sessão, as políticas de confiança para todas as funções conectadas ao provedor de identidade (IdP) que está passando etiquetas devem ter a permissão `sts:TagSession`. Para funções sem essa permissão na política de confiança, a operação `AssumeRole` falhará.
- Quando você solicita uma sessão, você pode especificar etiquetas de entidades de segurança como as etiquetas de sessão. As tags se aplicam às solicitações feitas usando as credenciais da sessão.
- As etiquetas de sessão são pares de chave-valor. Por exemplo, para adicionar informações de contato a uma sessão, você pode adicionar a chave de tag de sessão `email` e o valor da tag `johndoe@example.com`.
- As etiquetas de sessão devem seguir as regras para nomear etiquetas no IAM e no AWS STS. Este tópico inclui informações sobre diferenciação de maiúsculas e minúsculas e prefixos restritos que se aplicam às tags de sessão.
- As novas etiquetas de sessão substituem o perfil assumido existente ou as tags de sessão de usuário federado pela mesma chave de tag, independentemente da caixa alta ou baixa dos caracteres.
- Você não pode passar tags de sessão usando o Console de gerenciamento da AWS.
- As tags de sessão são válidas somente para a sessão atual.

- As tags de sessão oferecem suporte ao encadeamento de funções. Por padrão, o AWS STS não passa etiquetas para sessões de função subsequentes. No entanto, você pode definir tags de sessão como transitivas. As etiquetas transitivas persistem durante o encadeamento de funções e substituem os valores `ResourceTag` correspondentes após a avaliação da política de confiança de função. Para obter mais informações, consulte Encadeamento de funções com tags de sessão.
- Você pode usar tags de sessão para controlar o acesso a recursos ou para controlar quais tags podem ser passadas para uma sessão subsequente. Para obter mais informações, consulte Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC.
- Você pode visualizar as etiquetas de entidades de segurança da sua sessão, incluindo as etiquetas de sessão, nos logs do AWS CloudTrail. Para obter mais informações, consulte Visualizar etiquetas da sessão no CloudTrail.
- Você deve passar um único valor para cada etiqueta de sessão. O AWS STS não oferece suporte a etiquetas de sessão de vários valores.
- Você pode passar, no máximo, 50 tags de sessão. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.
- As políticas de sessão e as tags de sessão aumentam o tamanho do token da sessão. Se o token exceder o limite de tamanho, a AWS CLI ou a API da AWS retornará um erro. A mensagem de erro informa o tamanho do token da sessão e o tamanho máximo permitido, ambos em bytes. O elemento de resposta `SessionTokenUtilization` mostra qual porcentagem do limite de tamanho o token usa. O elemento de resposta `PackedPolicySize` está obsoleto. Use `SessionTokenUtilization` em vez disso.

Permissões necessárias para adicionar tags de sessão

Além da ação que corresponde à operação da API, é necessário ter a seguinte ação somente com permissão em sua política:

```
sts:TagSession
```

> **Important**
>
> Ao usar tags de sessão, as políticas de confiança da função para todas as funções conectadas a um provedor de identidade (IdP) devem ter a `sts:TagSession` permissão. A operação `AssumeRole` falhará em qualquer função conectada a um IdP que esteja

passando etiquetas de sessão sem essa permissão. Se você não quiser atualizar a política de confiança de função para cada função, use uma instância de IdP separada para passar as tags da sessão. Em seguida, adicione a permissão `sts:TagSession` apenas às funções conectadas ao IdP separado.

Você pode usar a ação `sts:TagSession` com as chaves de condição a seguir.

- `aws:PrincipalTag`: compara a etiqueta anexada à entidade de segurança que está fazendo a solicitação com a etiqueta que você especificou na política. Por exemplo, você pode permitir que um principal passe tags de sessão somente se o principal que está fazendo a solicitação tiver as tags especificadas.
- `aws:RequestTag`: compara o par de chave-valor da etiqueta passado na solicitação com o par de etiquetas que você especificou na política. Por exemplo, você pode permitir que o principal passe as tags de sessão especificadas, mas somente com os valores especificados.
- `aws:ResourceTag`: compara o par de chave-valor da etiqueta que você especificou na política com o par de chave-valor anexado ao recurso. Por exemplo, você pode permitir que a entidade de segurança passe etiquetas de sessão somente se a função que ela estiver assumindo incluir as etiquetas especificadas.
- `aws:TagKeys`: compara as chaves de etiqueta em uma solicitação com as chaves que você especificou na política. Por exemplo, você pode permitir que o principal passe apenas tags de sessão com as chaves de tag especificadas. Essa chave de condição limita o conjunto máximo de tags de sessão que podem ser passadas.
- `sts:TransitiveTagKeys`: compara as chaves de etiqueta de sessão transitiva na solicitação com aquelas especificadas na política. Por exemplo, você pode criar uma política para permitir que um principal defina apenas tags específicas como transitivas. As tags transitivas persistem durante o encadeamento de funções. Para obter mais informações, consulte Encadeamento de funções com tags de sessão.

Por exemplo, a política de confiança da função a seguir permite que o usuário `test-session-tags` assuma a função com a política anexada. Ao assumir a função, esse usuário deve usar a AWS CLI ou API da AWS para passar as três tags de sessão necessárias e o ID externo necessário. Além disso, o usuário pode optar por definir as tags `Project` e `Department` como transitivas.

Example Exemplo de política de confiança de função para tags de sessão

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-
tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
```

```
                    ]
                }
            }
        }
    ]
}
```

O que essa política faz?

- A instrução `AllowIamUserAssumeRole` permite que o usuário `test-session-tags` assuma a função com a política anexada. Ao assumir a função, esse usuário deve passar as tags de sessão necessárias e o ID externo.
  - O primeiro bloco condicional dessa instrução requer que o usuário passe as tags de sessão `Project`, `CostCenter` e `Department`. Os valores da etiqueta não importam nesta instrução, portanto, você pode usar caracteres curinga (*) para os valores de etiqueta. Esse bloqueio garante que o usuário passe pelo menos essas três etiquetas de sessão. Caso contrário, haverá falha na operação. O usuário pode passar tags adicionais.
  - O segundo bloco de condição requer que o usuário passe um ID externo com o valor `Example987`.
- A instrução `AllowPassSessionTagsAndTransitive` permite a ação somente com permissão `sts:TagSession`. Esta ação deve ser permitida antes que o usuário possa passar tags de sessão. Se sua política incluir a primeira instrução sem a segunda instrução, o usuário não poderá assumir a função.
  - O primeiro bloco de condição dessa instrução permite que o usuário passe qualquer valor para as tags de sessão `CostCenter` e `Project`. É possível fazer isso usando curingas (*) para o valor de tag na política, o que requer o uso do operador de condição StringLike .
  - O segundo bloco de condição permite que o usuário passe somente o valor `Engineering` ou `Marketing` para a tag de sessão `Department`.
  - O terceiro bloco de condição lista o conjunto máximo de etiquetas que você pode definir como transitivas. O usuário pode optar por definir um subconjunto ou nenhuma tag como transitiva. O usuário não pode definir etiquetas adicionais como transitivas. Você pode exigir que ele defina pelo menos uma das tags como transitivas adicionando outro bloco de condição que inclui `"Null":{"sts:TransitiveTagKeys":"false"}`.

Passar tags de sessão usando AssumeRole

A operação `AssumeRole` retorna um conjunto de credenciais temporárias que você pode usar para acessar recursos da AWS. Você pode usar o usuário do IAM ou credenciais de função para chamar `AssumeRole`. Para passar tags de sessão ao assumir uma função, use a opção `--tags` da AWS CLI ou o parâmetro `Tags` da API da AWS.

Para definir tags como transitivas, use a opção `--transitive-tag-keys` da AWS CLI ou o parâmetro `TransitiveTagKeys` da API da AWS. As tags transitivas persistem durante o encadeamento de funções. Para obter mais informações, consulte Encadeamento de funções com tags de sessão.

O exemplo a seguir mostra uma solicitação de exemplo que usa `AssumeRole`. Neste exemplo, ao assumir a função `my-role-example`, você cria uma sessão chamada `my-session`. Adicione os pares de chave-valor da tag de sessão `Project` = `Automation`, `CostCenter` = `12345` e `Department` = `Engineering`. Defina também as tags `Project` e `Department` como transitivas especificando suas chaves. Você deve passar um único valor para cada etiqueta de sessão. O AWS STS não oferece suporte a etiquetas de sessão de vários valores.

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Passar tags de sessão usando AssumeRoleWithSAML

A operação `AssumeRoleWithSAML` é autenticada com o uso de federação baseada em SAML. Essa operação retorna um conjunto de credenciais temporárias que você pode usar para acessar recursos da AWS. Para obter mais informações sobre como usar a federação baseada em SAML para acesso ao Console de gerenciamento da AWS, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS. Para obter detalhes sobre acesso à AWS CLI ou à API da AWS, consulte Federação SAML 2.0. Para obter um tutorial sobre como configurar a federação do SAML para seus usuários do Active Directory, consulte AWS Federated Authentication with Active Directory Federation Services (ADFS) no AWS Security Blog.

Como administrador, você pode permitir que os membros do diretório da empresa se agrupem na AWS usando a operação AWS STS da `AssumeRoleWithSAML`. Para isso, é necessário concluir as seguintes tarefas:

1. Configurar sua rede como um provedor SAML para a AWS.
2. Criar um provedor SAML no IAM
3. Criar um perfil para uma federação do SAML 2.0 (console)
4. Concluir a configuração do IdP SAML e criar declarações para a resposta de autenticação SAML

AWSA inclui provedores de identidade com experiência de ponta a ponta certificada para etiquetas de sessão com suas soluções de identidade. Para saber como usar esses provedores de identidade para configurar tags de sessão, consulte Integrar provedores de soluções SAML de terceiros com a AWS.

Para passar atributos SAML como tags de sessão, inclua o elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`*. Use o elemento `AttributeValue` para especificar o valor da tag. Inclua um elemento `Attribute` separado para cada tag de sessão.

Por exemplo, vamos supor que você queira passar os seguintes atributos de identidade como tags de sessão:

- `Project:Automation`
- `CostCenter:12345`
- `Department:Engineering`

Para passar esses atributos, inclua os seguintes elementos em sua declaração do SAML.

Example Exemplo de trecho de uma declaração SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
```

```
</Attribute>
```

Para definir as etiquetas anteriores como transitivas, inclua outro elemento `Attribute` com o atributo `Name` definido como `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. As tags transitivas persistem durante o encadeamento de funções. Para obter mais informações, consulte Encadeamento de funções com tags de sessão.

Para definir as etiquetas `Project` e `Department` como transitivas, use o seguinte atributo multivalor:

Example Exemplo de trecho de uma declaração SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Passar tags de sessão usando AssumeRoleWithWebIdentity

Use federação compatível com OpenID Connect (OIDC) para autenticar a operação `AssumeRoleWithWebIdentity`. Essa operação retorna um conjunto de credenciais temporárias que você pode usar para acessar recursos da AWS. Para obter mais informações sobre como usar a federação de identidades da web para Console de gerenciamento da AWS acesso, consulte Federação OIDC.

Para passar tags de sessão do OpenID Connect (OIDC), você deve incluir as tags de sessão no JSON Web Token (JWT) quando enviar a solicitação `AssumeRoleWithWebIdentity`. Para saber mais sobre tokens e reivindicações OIDC, consulte Usar tokens com grupos de usuários no Guia do desenvolvedor Amazon Cognito.

AWSA aceita dois formatos de declaração para inclusão de tags de sessão no JWT:

- Formato de declaração aninhada
- Formato de declaração nivelada

Formato de declaração aninhada

O formato de declaração aninhada usa uma estrutura dentro do namespace `https://aws.amazon.com/tags` no JWT. Nesse formato:

- As tags de entidade principal são representadas como um objeto aninhado sob a chave `principal_tags`.
- Cada tag de entidade principal é um valor de uma única string.
- As chaves de tag transitivas são representadas em uma matriz sob a chave `transitive_tag_keys`.
- Tanto `principal_tags` quanto `transitive_tag_keys` são aninhados sob o namespace `https://aws.amazon.com/tags`.

O exemplo a seguir mostra um JWT decodificado que usa o formato de objeto aninhado:

Example Exemplo de JSON Web Token decodificado que usa o formato de declaração aninhada

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

Formato de declaração nivelada

O formato de declaração nivelada é compatível com provedores de identidades que não aceitam objetos aninhados em declarações do JWT, como o Microsoft Entra ID. Nesse formato:

- As tags de entidade principal são representadas como declarações distintas com o prefixo `https://aws.amazon.com/tags/principal_tags/`.

- Cada tag de entidade principal é um valor de uma única string.
- As chaves de tag transitivas são representadas em uma única declaração como uma matriz de strings com o prefixo `https://aws.amazon.com/tags/transitive_tag_keys`.

Agora, vamos ver como as mesmas informações são representadas usando o formato de declaração nivelada:

Example Exemplo de JSON Web Token decodificado que usa o formato de declaração nivelada

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags/principal_tags/Project": "Automation",
    "https://aws.amazon.com/tags/principal_tags/CostCenter": "987654",
    "https://aws.amazon.com/tags/principal_tags/Department": "Engineering",
    "https://aws.amazon.com/tags/transitive_tag_keys": [
        "Project",
        "CostCenter"
    ]
}
```

Ambos os exemplos de JWT decodificados mostram uma chamada para `AssumeRoleWithWebIdentity` com as tags de sessão `Project`, `CostCenter` e `Department`. Ambos os tokens definem as tags `Project` e `CostCenter` como transitivas. As tags transitivas persistem durante o encadeamento de funções. Para obter mais informações, consulte Encadeamento de funções com tags de sessão.

O formato de declaração nivelada alcança o mesmo resultado que o formato de declaração aninhada, mas usa uma estrutura nivelada para tags. Ele permite que você inclua tags de sessão em ambientes em que objetos JSON aninhados não são aceitos em declarações do JWT. Ao usar qualquer um dos formatos, certifique-se de que seu provedor de identidades esteja configurado para emitir tokens com as estruturas de declaração apropriadas. A AWS aceita os dois formatos de declaração para você poder escolher o que melhor se adapta aos requisitos específicos do seu provedor de identidades.

Passar tags de sessão usando GetFederationToken

`GetFederationToken` permite federar seu usuário. Essa operação retorna um conjunto de credenciais temporárias que você pode usar para acessar recursos da AWS. Para adicionar tags à sua sessão de usuário federado, use a opção `--tags` da AWS CLI ou o parâmetro `Tags` da API da AWS. Não é possível definir etiquetas de sessão como transitivas ao usar o`GetFederationToken` porque você não pode usar as credenciais temporárias para assumir uma função. Não é possível usar encadeamento de função neste caso.

O exemplo a seguir mostra uma solicitação de exemplo usando `GetFederationToken`. Neste exemplo, ao solicitar o token, você cria uma sessão chamada `my-fed-user`. Adicione os pares de chave-valor da tag de sessão `Project` = `Automation` e `Department` = `Engineering`.

Example Exemplo de solicitação da CLI GetFederationToken

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Quando você usa as credenciais temporárias retornadas pela operação `GetFederationToken`, as etiquetas da entidade de segurança da sessão incluem as etiquetas do usuário e as etiquetas passadas da sessão.

Encadeamento de funções com tags de sessão

Você pode assumir uma função e usar as credenciais temporárias para assumir outra função. É possível continuar de sessão em sessão. Isso é chamado de encadeamento de funções. Ao passar tags de sessão enquanto assume uma função, você pode definir as chaves como transitivas. Isso garante que essas tags de sessão passem para sessões subsequentes em uma cadeia de funções. Não é possível definir tags de função como transitivas. Para passar essas tags para sessões subsequentes, especifique-as como tags de sessão.

> **Note**
>
> As etiquetas transitivas persistem durante o encadeamento de funções e substituem os valores `ResourceTag` correspondentes após a avaliação da política de confiança de função.

O exemplo a seguir mostra como o AWS STS passa etiquetas de sessão, etiquetas transitivas e etiquetas de função em sessões subsequentes em uma cadeia de função.

Neste exemplo de cenário de encadeamento de funções, você usa uma chave de acesso de usuário do IAM na AWS CLI para assumir uma função chamada `Role1`. Use as credenciais de sessão resultantes para assumir uma segunda função chamada `Role2`. Você poderá usar as credenciais de segunda sessão para assumir uma terceira função chamada `Role3`. Essas solicitações ocorrem como três operações distintas. Cada função já está etiquetada no IAM. E durante cada solicitação, você passa tags de sessão adicionais.
Encadeamento de funções.

Ao encadear funções, você pode garantir que as tags de uma sessão anterior persistam para as sessões posteriores. Para fazer isso usando o comando da CLI `assume-role`, é necessário passar a etiqueta como etiqueta de sessão e definir a etiqueta como transitiva. Passe a tag `Star = 1` como tag de sessão. O comando também anexa a etiqueta `Heart = 1` à função e se aplica como uma etiqueta de entidade de segurança quando você usa a sessão. No entanto, você também deseja que a tag `Heart = 1` passe automaticamente para a segunda ou terceira sessão. Para isso, inclua a tag manualmente como tag de sessão. As etiquetas de entidade de segurança de sessão resultantes incluem ambas as etiquetas e as define como transitivas.
Assumir a primeira função em uma cadeia de perfis.

Execute essa solicitação usando o seguinte comando da AWS CLI:

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

Use as credenciais dessa sessão para assumir `Role2`. O comando anexa a etiqueta `Sun = 2` à segunda função e se aplica como uma etiqueta de entidade de segurança quando você usa a segunda sessão. As tags `Heart` e `Star` herdam as tags transitivas de sessão na primeira sessão. As etiquetas de entidade de segurança resultantes da segunda sessão são `Heart = 1`, `Star = 1` e `Sun = 2`. `Heart` e `Star` continuam a ser transitivas. A etiqueta `Sun` anexada à `Role2` não é marcada como transitiva porque não é uma etiqueta de sessão. Sessões futuras não herdam esta etiqueta.
Assumir a segunda função em uma cadeia de perfis.

Execute essa segunda solicitação usando o seguinte comando da AWS CLI:

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

Use as credenciais da segunda sessão para assumir `Role3`. As tags principais da terceira sessão vêm de quaisquer novas tags de sessão, as tags de sessão transitivas herdadas e as tags de função. As etiquetas `Heart = 1` e `Star = 1` na segunda sessão são herdadas da etiqueta de sessão transitiva na primeira sessão. Se você tentar passar a etiqueta de sessão `Sun = 2`, a operação falhará. A etiqueta de sessão `Star = 1` herdada substitui a etiqueta de função `Star = 3`. No encadeamento de funções, o valor de uma etiqueta transitiva substitui a função correspondente ao valor `ResourceTag` após a avaliação da política de confiança de função. Neste exemplo, se `Role3` usar `Star` como `ResourceTag` na política de confiança de função e definir `ResourceTag` como o valor de etiqueta transitiva da sessão de função de chamada. A tag `Lightning` do perfil também se aplica à terceira sessão e não é definida como transitiva.
Assumir a terceira função em uma cadeia de perfis.

Execute a terceira solicitação usando o seguinte comando da AWS CLI:

Example Exemplo de solicitação da CLI AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Usar tags de sessão para ABAC

O controle de acesso baseado em atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos da etiqueta.

Se sua empresa usa um provedor de identidade (IdP) baseado em OIDC ou SAML para gerenciar identidades de usuário, você pode configurar sua declaração para passar etiquetas de sessão para a AWS. Por exemplo, com identidades de usuários corporativos, quando seus funcionários fazem federação com a AWS, a AWS aplica seus atributos às respectivas entidades de segurança resultantes. Você pode usar o ABAC para conceder ou não permissões com base nesses atributos. Para obter detalhes, consulte Tutorial do IAM: Usar etiquetas de sessão SAML para ABAC.

Para obter mais informações sobre como usar o IAM Identity Center com ABAC, consulte Attributes for access control (Atributos para o controle de acesso) no Guia do Usuário do Centro de Identidade do AWS IAM.

Visualizar etiquetas da sessão no CloudTrail

Você pode usar AWS CloudTrail para visualizar as solicitações usadas para assumir funções ou federar usuários. O arquivo de log do CloudTrail inclui informações sobre as etiquetas de entidade de segurança para a função assumida ou a sessão de usuário federado. Para obter mais informações, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.

Por exemplo, vamos supor que você faça uma solicitação AWS STS do `AssumeRoleWithSAML`, passe tags de sessão e defina essas tags como transitivas. Você pode encontrar as seguintes informações em seu log do CloudTrail.

Example Exemplo de log AssumeRoleWithSAML do CloudTrail

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

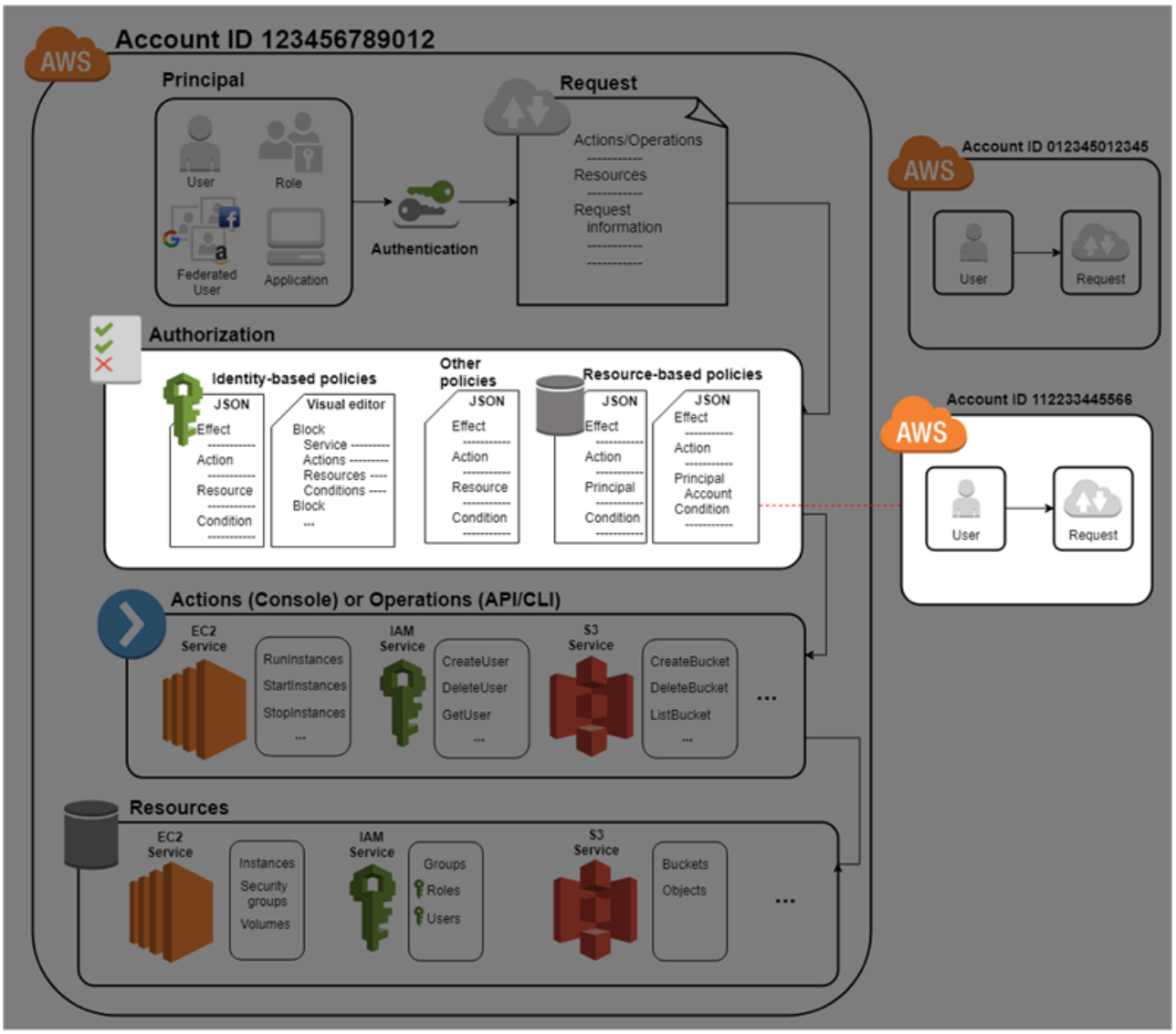
Você pode visualizar os logs do CloudTrail a seguir para exibir eventos que usam etiquetas de sessão.

- Exemplo de evento de API de encadeamento de funções do AWS STS no arquivo de log do CloudTrail
- Exemplo de evento de API SAML do AWS STS no arquivo de log do CloudTrail
- Exemplo de evento da API AWS STS do OIDC no arquivo de log do CloudTrail

Gerenciamento de acesso para recursos da AWS

O AWS Identity and Access Management (IAM) é um serviço da Web que ajuda você a controlar o acesso aos recursos da AWS de forma segura. Quando um principal faz uma solicitação no AWS, o código de aplicação do AWS verifica se o principal está autenticado (fez login) e autorizado (tem permissões). Você gerencia o acesso na AWS criando políticas e anexando-as às identidades do IAM ou aos recursos da AWS. As políticas são documentos JSON no AWS que, quando anexadas a uma identidade ou recurso, definem suas permissões. Para obter mais informações sobre os tipos e os usos de políticas, consulte Políticas e permissões no AWS Identity and Access Management.

Para obter mais detalhes sobre o restante do processo de autenticação e autorização, consulte Como o IAM funciona.



Ao fazer uma autorização, o código de aplicação do AWS usa os valores do contexto da solicitação para buscar as políticas correspondentes e determinar se ela será permitida ou negada.

AWSO verifica cada política que se aplica ao contexto da solicitação. Se uma única política negar a solicitação, a AWS negará toda a solicitação e interromperá a avaliação de políticas. Esse processo é chamado de negação explícita. Como as solicitações são negadas por padrão, o IAM autorizará sua solicitação apenas se todas as partes de sua solicitação forem permitidas pelas políticas aplicáveis. A lógica de avaliação para uma solicitação em uma única conta segue estas regras:

- Por padrão, todas as solicitações são implicitamente negadas. (Opcionalmente, por padrão, Usuário raiz da conta da AWS tem acesso total.)
- Uma permissão explícita em uma política baseada em recurso ou identidade substitui esse padrão.
- Se houver um limite de permissões, SCP de AWS Organizations ou política de sessão, isso poderá substituir a permissão com uma negação implícita.
- Uma negação explícita em qualquer política substitui todas as permissões.

Depois que sua solicitação for autenticada e autorizada, a AWS aprovará a solicitação. Se você precisar fazer uma solicitação em uma conta diferente, uma política na outra conta deverá permitir que você acesse o recurso. Além disso, a entidade do IAM que você usa para fazer a solicitação deve ter uma política baseada em identidade que permita a solicitação.

Você também pode usar Delegação temporária do IAM para conceder à Amazon e aos parceiros da AWS o acesso limitado e com limite de tempo para configurar os serviços da AWS na sua conta.

Recursos de gerenciamento de acesso

Para obter mais informações sobre permissões e criação de políticas, consulte os seguintes recursos:

Os registroa a seguir no AWS Security Blog abrangem formas comuns de gravar políticas para acesso a buckets e objetos do Amazon S3.

- Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket
- Writing IAM policies: Grant Access to User-Specific Folders in an Amazon S3 Bucket
- IAM Policies and Bucket Policies and ACLs! Nossa! (Controlar o acesso aos recursos do S3)
- Uma introdução às permissões em nível de recursos do RDS
- Desmistificação de permissões em nível de recursos do EC2

Políticas e permissões no AWS Identity and Access Management

Gerencie o acesso na AWS criando políticas e anexando-as a identidades do IAM (usuários, grupos de usuários ou perfis) ou a recursos da AWS. Uma política é um objeto na AWS que, quando associado a uma identidade ou um recurso, define suas permissões. A AWS avalia essas políticas quando uma entidade principal do IAM (usuário ou perfil) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas são armazenadas na AWS como documentos JSON. A AWS oferece suporte a nove tipos de políticas: políticas baseadas em identidade, políticas baseadas em recursos, políticas de endpoints da VPC, limites de permissões, políticas de controle de serviços (SCPs) do AWS Organizations, políticas de controle de recursos (RCPs) do AWS Organizations, listas de controle de acesso (ACL), compartilhamentos do Resource Access Manager (RAM) e políticas de sessão.

As políticas do IAM definem permissões para uma ação independentemente do método usado para executar a operação. Por exemplo, se uma política permitir a ação GetUser, um usuário com essa política poderá obter informações de usuários no Console de gerenciamento da AWS, na AWS CLI ou na API da AWS. Ao criar um usuário do IAM, você pode optar por permitir acesso ao console ou programático. Se o acesso ao console for permitido, o usuário do IAM poderá fazer login nele usando suas credenciais. Se o acesso programático for permitido, o usuário poderá usar as chaves de acesso para trabalhar com a CLI ou a API.

Tipos de políticas

Os seguintes tipos de políticas, listados em ordem do usado com mais frequência a menos usado, estão disponíveis para uso na AWS. Para obter mais detalhes, consulte as seções a seguir para cada tipo de política.

- Políticas baseadas em identidade: anexe políticas gerenciadas e em linha a identidades do IAM (usuários, grupos aos quais os usuários pertencem ou funções). As políticas baseadas em identidade concedem permissões a uma identidade.
- Políticas baseadas em recurso: anexe políticas em linha a recursos. Os exemplos de políticas baseadas em recurso mais comuns são as políticas de bucket do Amazon S3 e as políticas de confiança de funções do IAM. As políticas baseadas em recurso concedem permissões à entidade principal que é especificada na política. As entidades principais podem ser na mesma conta como o recurso ou em outras contas.
- Políticas de endpoint da VPC: anexe a um endpoint da VPC para controlar quais entidades principais podem usar as políticas de endpoint da VPC e quais recursos podem ser acessados por

meio dele. As políticas de endpoint da VPC atuam como um limite de acesso adicional com escopo para o tráfego que atravessa o endpoint.

- Limites de permissões: use uma política gerenciada como o limite de permissões para uma entidade do IAM (usuário ou função). Essa política define o número máximo de permissões que as políticas baseadas em identidade podem conceder a uma entidade, mas não concede permissões. Os limites de permissões não definem o número máximo de permissões que uma política baseada em recurso pode conceder a uma entidade.
- SCPs do AWS Organizations: use uma política de controle de serviço (SCP) do AWS Organizations para definir o número máximo de permissões para usuários do IAM e perfis do IAM nas contas da sua organização ou unidade organizacional (UO). As SCPs limitam as permissões que as políticas baseadas em identidade ou políticas baseadas em recurso concedem a usuários do IAM ou perfis do IAM na conta. As SCPs não concedem permissões.
- RCPs do AWS Organizations: use uma política de controle de recurso (RCP) do AWS Organizations para definir o número máximo de permissões para recursos nas contas da sua organização ou unidade organizacional (UO). As RCPs limitam as permissões que as políticas baseadas em identidade e recursos podem conceder aos recursos nas contas da sua organização. As RCPs não concedem permissões.
- Listas de controle de acesso (ACLs)): use ACLs para controlar quais entidades de segurança em outras contas podem acessar o recurso ao qual a ACL está anexada. As ACLs são semelhantes às políticas baseadas em recurso, embora sejam o único tipo de política que não usa a estrutura de documento de política JSON. As ACLs são políticas de permissões entre contas que concedem permissões para a entidade principal especificada. As ACLs não podem conceder permissões para entidades na mesma conta.
- Compartilhamentos de recursos do AWS RAM: use o AWS Resource Access Manager para compartilhar recursos entre unidades organizacionais de Contas da AWS ou uma organização inteira sem escrever políticas individuais baseadas em recursos para cada recurso compartilhado.
- Políticas de sessão: transmita políticas de sessão avançadas ao usar a AWS CLI ou a API da AWS para assumir uma função ou um usuário federado. As políticas de sessão limitam as permissões que as políticas baseada em identidade do usuário ou função concedem à sessão. As políticas de sessão limitam as permissões para uma sessão criada, mas não concedem permissões. Para obter mais informações, consulte Políticas de sessão.

Políticas baseadas em identidade

As Políticas baseadas em identidade são documentos de política de permissões JSON que controlam quais ações uma identidade (usuários, grupos de usuários e funções) pode realizar, em quais recursos e em que condições. As políticas baseadas em identidade podem ser categorizadas em:

- Políticas gerenciadas: políticas autônomas baseadas em identidade que você pode anexar a vários usuários, grupos e perfis na Conta da AWS. Existem dois tipos de políticas gerenciadas:
  - Políticas gerenciadas pela AWS: políticas gerenciadas que são criadas e gerenciadas pela AWS.
  - Políticas gerenciadas pelo cliente: políticas gerenciadas que você criar e gerenciar na sua Conta da AWS. As políticas gerenciadas pelo cliente oferecem um controle mais preciso de suas políticas do que as políticas gerenciadas pela AWS.
- Políticas em linha: políticas adicionadas diretamente a um único usuário, grupo ou perfil. As políticas em linha mantêm um relacionamento estrito de um para um entre uma política e uma identidade. Elas são excluídas quando você exclui a identidade.

Para saber como escolher entre políticas gerenciadas e em linha, consulte Escolher entre políticas gerenciadas e políticas em linha.

Políticas baseadas em recursos

Políticas baseadas em recurso são documentos de política JSON que você anexa a um recurso, como um bucket do Amazon S3. Essas políticas concedem permissão à entidade principal especificada para executar ações específicas nesse recurso e definem em que condições isso se aplica. As políticas baseadas em recursos são políticas em linha. Não há políticas baseadas em recursos gerenciadas.

Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em recursos. Adicionar uma entidade principal entre contas à política baseada em recurso é apenas metade da tarefa de estabelecimento da relação de confiança. Quando a entidade principal e o recurso estiverem em Contas da AWS separadas, também será necessário usar uma política baseada em identidade para conceder o acesso a essa entidade principal para o recurso. No entanto, se uma política baseada em recurso conceder acesso a uma entidade principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Para obter instruções detalhadas sobre a concessão de acesso

entre serviços, consulte Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM.

O serviço do IAM oferece suporte a apenas um tipo de política baseada em recurso chamada política de confiança de uma função, que é anexada a uma função do IAM. Uma função do IAM é uma identidade e um recurso que é compatível com as políticas baseadas em recurso. Por esse motivo, você deve anexar uma política de confiança e uma política baseada em identidade a uma função do IAM. As políticas de confiança definem quais entidades principais (contas, usuários, funções e usuários federados) podem assumir a função. Para saber como as funções do IAM diferem de outras políticas baseadas em recurso, consulte Acesso a recursos entre contas no IAM.

Para ver quais outros serviços oferecem suporte a políticas baseadas em recurso, consulte AWSServiços da que funcionam com o IAM. Para saber mais sobre as políticas baseadas em recursos, consulte Políticas baseadas em identidade e em recurso. Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

Políticas de VPC endpoint

Uma política de endpoint da VPC é uma política de recurso que pode ser anexada a um endpoint da VPC para controlar quais entidades principais podem usar o endpoint e quais recursos podem ser acessados por meio dele. As políticas de endpoint da VPC não anulam nem substituem as políticas baseadas em identidade ou políticas baseadas em recursos vinculadas ao serviço de destino. Elas atuam como um limite de acesso adicional ao tráfego que atravessa o endpoint.

As políticas de endpoint da VPC são usadas para garantir que somente identidades confiáveis em sua organização do AWS Organizations possam acessar recursos confiáveis por meio de seus endpoints da VPC. Se você não anexar uma política de endpoint personalizada, AWS anexará uma política padrão para permitir acesso total. Para obter mais informações sobre as políticas de endpoint da VPC, consulte Controlar o acesso aos endpoints da VPC usando políticas de endpoint no Guia do AWS PrivateLink.

Limites de permissões do IAM

Um limite de permissões é um recurso avançado no qual você define as permissões máximas que uma política baseada em identidade pode conceder a uma entidade do IAM. Quando você definir um limite de permissões para uma entidade, ela poderá executar apenas as ações que são permitidas por ambas as políticas baseadas em identidade e seus limites de permissões.

Se você especificar um perfil, sessão ou usuário no elemento da entidade principal de uma política baseada em recursos, não será necessária uma permissão explícita no limite de permissão. Contudo, se você especificar um ARN de perfil no elemento da entidade principal de uma política baseada em recursos, será necessária uma permissão explícita no limite de permissão. Em ambos os casos, uma negação explícita no limite da permissão é efetiva. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre esses limites de permissões, consulte Limites de permissões para entidades do IAM.

AWS OrganizationsPolíticas de controle de serviço (SCPs) do

Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. As SCPs são políticas JSON que especificam o máximo de permissões para usuários do IAM e perfis do IAM nas contas de uma organização ou unidade organizacional (UO). A SCP limita as permissões para entidades principais em contas-membro, o que inclui cada Usuário raiz da conta da AWS. Uma negação explícita em qualquer uma dessas políticas se sobrepõe a uma permissão nas outras políticas.

Para obter mais informações sobre o AWS Organizations e as SCPs, consulte Políticas de controle de serviço (SCP) no Guia do usuário do AWS Organizations.

AWS OrganizationsPolíticas de controle de recursos (RCPs) do

Se você habilitar todos os atributos em uma organização, poderá usar políticas de controle de recursos (RCPs) para aplicar centralmente controles de acesso a recursos em várias Contas da AWS. As RCPs são políticas JSON que podem ser usadas para definir o máximo de permissões disponíveis para recursos em suas contas sem atualizar as políticas do IAM anexadas a cada recurso que você possui. A RCP limita as permissões para recursos nas contas-membro e pode afetar as permissões efetivas para identidades, incluindo o Usuário raiz da conta da AWS, independentemente de pertencerem a sua organização. Uma negação explícita em qualquer RCP aplicável substitui uma permissão em outras políticas que possam estar associadas a identidades ou recursos individuais.

Para obter mais informações sobre o AWS Organizations e as RCPs, incluindo uma lista dos Serviços da AWS que oferecem suporte às RCPs, consulte Políticas de controle de recursos (RCPs) no Guia do usuário do AWS Organizations.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) são políticas de serviço que permitem controlar quais entidades principais em outra conta podem acessar um recurso. As ACLs não podem ser usadas

para controlar o acesso a um principal na mesma conta. As ACLs são semelhantes às políticas baseadas em recurso, embora sejam o único tipo de política que não usa o formato de documento de política JSON. Amazon S3, AWS WAF e Amazon VPC são exemplos de serviços que oferecem compatibilidade com ACLs. Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do desenvolvedor do Amazon Simple Storage Service.

Compartilhamentos de recursos do AWS Resource Manager (AWS RAM)

O AWS Resource Access Manager (AWS RAM) permite compartilhar recursos criados em uma Conta da AWS com outras Contas da AWS, unidades organizacionais ou uma organização inteira no AWS Organizations sem precisar escrever políticas individuais baseadas em recursos para cada recurso compartilhado. Embora você possa conceder acesso entre contas anexando uma política baseada em recursos diretamente a um recurso, o AWS RAM fornece uma alternativa gerenciada e centralizada que elimina a necessidade de enumerar IDs de conta em políticas ou manter documentos de política idênticos em todas as contas.

Com o RAM, as contas consumidoras veem os recursos compartilhados de forma nativa em seus consoles de serviço, os proprietários dos recursos mantêm a propriedade total e a visibilidade de quem tem acesso, e as permissões gerenciadas definem o máximo de ações que os consumidores podem realizar, tudo governado por um único compartilhamento de recursos, em vez de políticas por recurso. Os casos de uso comuns incluem o compartilhamento de sub-redes VPC, anexos do AWS Transit Gateway, regras do Route 53 Resolver e configurações do License Manager entre contas. Para obter mais informações, consulte O que é o AWS Resource Access Manager? no Guia do usuário do AWS Resource Access Manager.

Políticas de sessão

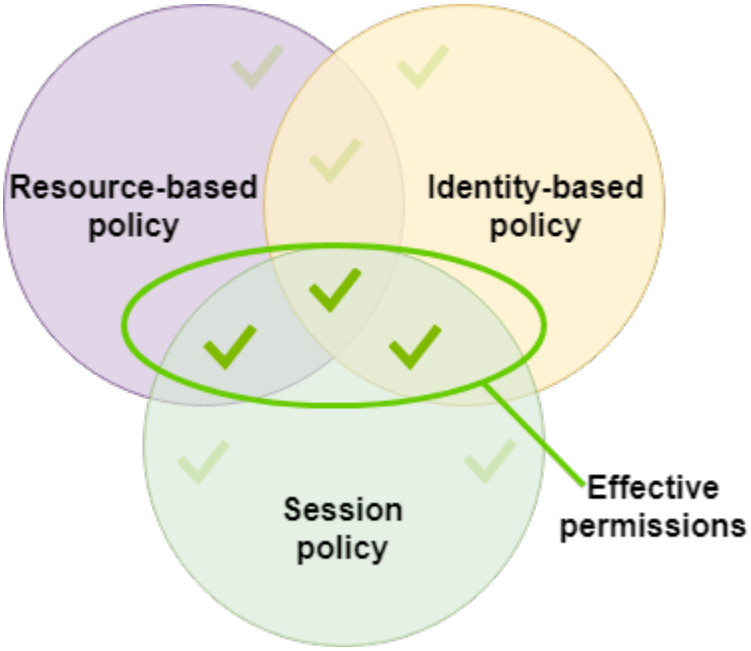
As políticas de sessão são políticas avançadas que você transmite como um parâmetro quando você cria de forma programática uma sessão temporária para um perfil ou uma entidade principal de usuário federado do AWS STS. As permissões para uma sessão são a interseção das políticas baseadas em identidade para a entidade (usuário ou função) do IAM usada para criar a sessão e as políticas da sessão. As permissões também podem ser provenientes de uma política baseada em recursos. Uma negação explícita em qualquer uma dessas políticas substitui a permissão.

É possível criar uma sessão de perfil e transmitir políticas de sessão de forma programática usando as operações de API `AssumeRole`, `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity`. É possível passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. É possível usar o parâmetro `PolicyArns` para especificar até 10 políticas de sessão

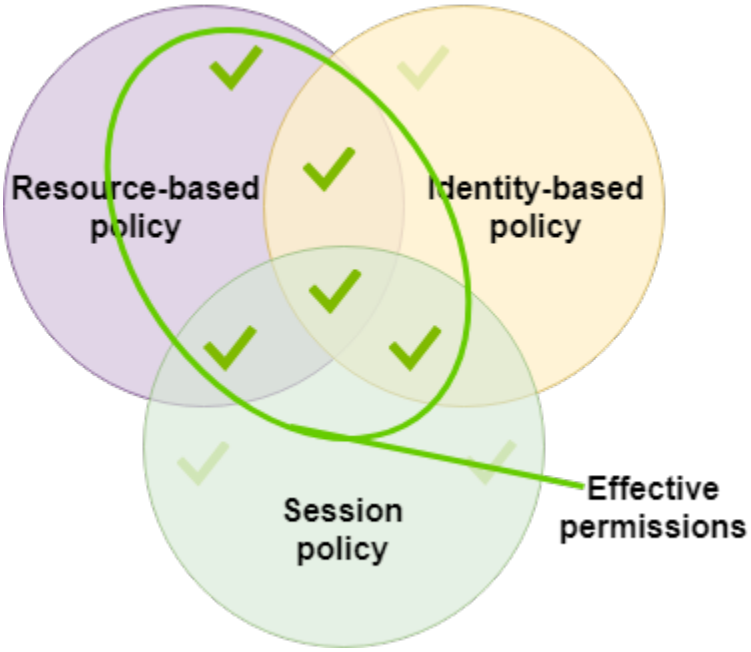
gerenciadas. Para obter mais informações sobre a criação de uma sessão de função, consulte Permissões de credenciais de segurança temporárias.

Ao criar uma sessão de entidade principal de usuário federado do AWS STS, use as chaves de acesso do usuário do IAM para chamar de forma programática a operação da API `GetFederationToken`. Você também deve transmitir as políticas da sessão. As permissões da sessão resultam da interseção da política baseada em identidade do usuário e da política de sessão. Para obter mais informações sobre a criação de uma sessão de usuário federado, consulte Solicitar credenciais por meio de um intermediador de identidades personalizado.

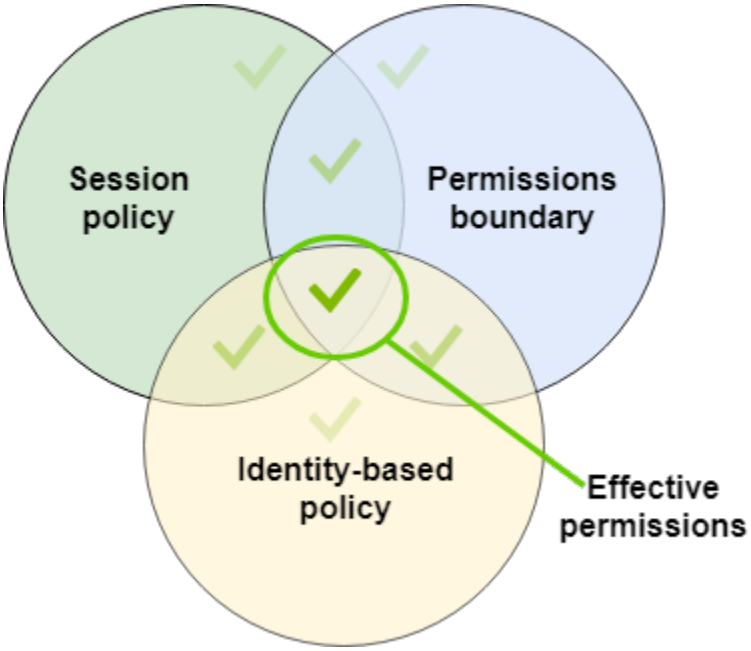
Uma política baseada em recurso pode especificar o ARN do usuário ou função como uma entidade principal. Nesse caso, as permissões da política baseada em recurso são adicionadas à função ou política baseada em identidade do usuário antes que a sessão seja criada. A política de sessão limita o total de permissões concedidas pela política baseada em recurso e política baseada em identidade. As permissões da sessão resultante são a interseção das políticas de sessão e das políticas baseadas em recurso, mais a interseção das políticas de sessão e das políticas baseadas em identidade.



Uma política baseada em recurso pode especificar o ARN da sessão como uma entidade principal. Nesse caso, as permissões da política baseada em recurso são adicionadas depois que a sessão for criada. As permissões da política baseada em recurso não são limitadas pela política de sessão. A sessão resultante tem todas as permissões da política baseada em recurso além da interseção da política baseada em identidade e da política de sessão.



Um limite de permissões pode definir o número máximo de permissões para um usuário ou uma função que é usada para criar uma sessão. Nesse caso, as permissões da sessão resultam da interseção da política de sessão, do limite de permissões e da política baseada em identidade. No entanto, um limite de permissões não limita as permissões concedidas por uma política baseada em recurso que especifica o ARN da sessão resultante.



Políticas e o usuário-raiz

O Usuário raiz da conta da AWS é afetado por alguns tipos de políticas, mas não por outros. Você não pode anexar políticas baseadas em identidade ao usuário-raiz e não pode definir o limite de permissões para o usuário-raiz. No entanto, você pode especificar o usuário-raiz como a entidade principal em uma política baseada em recurso ou uma ACL. Um usuário-raiz ainda é o membro de uma conta. Se essa conta for membro de uma organização no AWS Organizations, o usuário-raiz será afetado por SCPs e RCPs para a conta.

Visão geral das políticas de JSON

A maioria das políticas é armazenada na AWS como documentos JSON. As políticas baseadas em identidade e as políticas usadas para definir limites de permissões são documentos de política JSON que você anexa a um usuário ou a uma função. Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. As SCPs e RCPs são documentos de políticas JSON com sintaxe restrita que você anexa a uma raiz de organização do AWS Organizations, unidade organizacional (UO) ou conta. As ACLs também são anexadas a um recurso, mas você deve usar uma sintaxe diferente. As políticas de sessão são políticas JSON que você fornece ao assumir uma sessão de função ou usuário federado.

Você não precisa compreender a sintaxe JSON. É possível usar o editor visual no Console de gerenciamento da AWS para criar e editar políticas gerenciadas pelo cliente sem nunca ter usado o JSON. No entanto, se você usar políticas em linha para grupos ou políticas complexas, ainda será necessário criar e editar essas políticas no editor de JSON usando o console. Para obter informações sobre como usar o editor visual, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente e Editar políticas do IAM.

Quando você cria ou edita uma política JSON, o IAM pode executar a validação de políticas para ajudar você a criar uma política eficaz. O IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de política do IAM. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer.

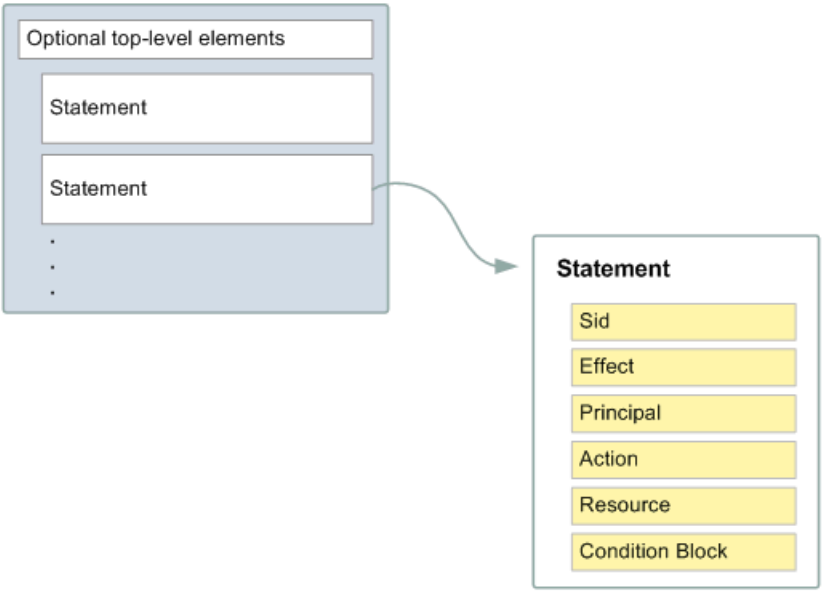
Estrutura de documento de política JSON

Como mostrado na figura a seguir, um documento de política JSON inclui estes elementos:

- Informações opcionais da política na parte superior do documento

- Uma ou mais instruções individuais

Cada instrução inclui informações sobre uma única permissão. Se uma política incluir várias instruções, a AWS aplicará um `OR` lógico a todas as instruções ao avaliá-las. Se várias políticas se aplicarem a uma solicitação, a AWS aplicará um `OR` lógico a todas essas políticas ao avaliá-las.



As informações em uma instrução estão contidas em uma série de elementos.

- Version – especifique a versão do idioma da política que deseja usar. É recomendável usar a versão `2012-10-17` mais recente. Para obter mais informações, consulte . Elementos de política JSON do IAM: Version
- Statement – use este elemento de política principal como um contêiner para os elementos a seguir. É possível incluir mais de uma instrução em uma política.
- Sid (opcional): inclua um ID de instrução opcional para diferenciar entre suas instruções.
- Effect: use `Allow` ou `Deny` para indicar se a política permite ou nega acesso.
- Principal (obrigatório em algumas circunstâncias): se você criar uma política baseada em recurso, deverá indicar a conta, o usuário, o perfil ou a entidade principal do usuário federado do AWS STS ao qual deseja permitir ou negar acesso. Se estiver criando uma política de permissões do IAM para anexar a um usuário ou a uma função, você não poderá incluir esse elemento. A entidade principal é implícita como esse usuário ou função.
- Action – inclua uma lista de ações que a política permite ou nega.

- Resource (obrigatório em algumas circunstâncias): se você criar uma política de permissões do IAM, deverá especificar uma lista de recursos aos quais as ações se aplicam. Se você criar uma política baseada em recursos, isso dependerá do recurso que você está usando para determinar se esse elemento é necessário ou não.
- Condition (opcional): especifique as circunstâncias sob as quais a política concede a permissão.

Para saber mais sobre esses e outros elementos mais avançados de políticas, consulte Referência de elemento de política JSON do IAM.

Várias instruções e várias políticas

Se desejar definir mais de uma permissão para uma entidade (usuário ou função), você poderá usar várias instruções em uma única política. Você também pode anexar várias políticas. Se você tentar definir várias permissões em uma única instrução, sua política poderá não conceder o acesso esperado. É recomendável dividir políticas por tipo de recurso.

Devido ao tamanho limitado das políticas, poderá ser necessário usar várias políticas para permissões mais complexas. Também é uma boa ideia criar agrupamentos funcionais de permissões em uma política gerenciada pelo cliente separada. Por exemplo, crie uma política para gerenciamento de usuários do IAM, uma para autogerenciamento e outra política para gerenciamento de buckets do S3. Independentemente da combinação de várias instruções e de várias políticas, a AWS avalia suas políticas da mesma forma.

Por exemplo, a política a seguir tem três instruções, cada uma delas define um conjunto separado de permissões em uma única conta. As instruções definem o seguinte:

- A primeira instrução, com um `Sid` (ID de instrução) de `FirstStatement`, permite que o usuário com a política anexada altere sua própria senha. O elemento `Resource` nessa instrução é "*" (o que significa "todos os recursos"). No entanto, na prática, a operação de API `ChangePassword` (ou o comando da CLI `change-password` equivalente) afeta somente a senha do usuário que faz a solicitação.
- A segunda instrução permite que o usuário liste todos os buckets do Amazon S3 na sua Conta da AWS. O elemento `Resource` nessa instrução é "*" (o que significa "todos os recursos"). Porém, como as políticas não concedem acesso aos recursos em outras contas, o usuário pode listar somente os buckets na sua própria Conta da AWS.
- A terceira instrução permite que o usuário liste e recupere qualquer objeto que esteja em um bucket chamado `amzn-s3-demo-bucket-confidential-data`, mas somente quando o

usuário estiver autenticado com autenticação multifator (MFA). O elemento `Condition` na política impõe a autenticação MFA.

Quando uma instrução de política contém um elemento `Condition`, ela só entrará em vigor quando o elemento `Condition` for verdadeiro. Nesse caso, a `Condition` é avaliada como verdadeira quando o usuário é autenticado por MFA. Se o usuário não for autenticado por MFA, essa `Condition` será avaliada como falsa. Nesse caso, a terceira instrução dessa política não se aplicará, e o usuário não terá acesso ao bucket `amzn-s3-demo-bucket-confidential-data`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data",
        "arn:aws:s3:::amzn-s3-demo-bucket-confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
```

```
}
```

Exemplos de sintaxe de política JSON

A seguinte política baseada em identidade permite que a entidade principal implícita liste um único bucket do Amazon S3 chamado `amzn-s3-demo-bucket`:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
  }
}
```

A seguinte política baseada em recurso pode ser anexada a um bucket do Amazon S3. A política permite que membros de uma Conta da AWS específica execute qualquer ação do Amazon S3 no bucket denominado `amzn-s3-demo-bucket`. Ela permite qualquer ação que pode ser executada em um bucket ou em seus objetos. (Como a política concede confiança apenas à conta, os usuários individuais da conta ainda devem receber permissões para as ações especificadas do Amazon S3.)

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": "s3:*",
```

```
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Para visualizar políticas de exemplo para cenários comuns, consulte Exemplos de políticas baseadas em identidade do IAM.

Conceder privilégio mínimo

Ao criar políticas do IAM, siga a orientação de segurança padrão de concessão de privilégio mínimo ou conceda apenas as permissões necessárias para realizar uma tarefa. Determine o que os usuários e as funções precisam fazer e, em seguida, crie políticas que permitam que eles executem apenas essas tarefas.

Comece com um conjunto mínimo de permissões e conceda permissões adicionais conforme necessário. Fazer isso é mais seguro do que começar com permissões que são muito lenientes e tentar restringi-las superiormente.

Como alternativa ao privilégio mínimo, você pode usar políticas gerenciadas pela AWS ou políticas com permissões com curinga * para iniciar as políticas. Considere o risco de segurança de conceder às suas entidades de segurança mais permissões do que elas precisam para realizar um trabalho. Monitore esses entidades de segurança para saber quais permissões elas estão usando. Em seguida, escreva políticas de privilégio mínimos.

O IAM fornece várias opções para ajudar você a refinar as permissões concedidas.

- Compreender agrupamentos de nível de acesso: você pode usar agrupamentos no nível de acesso para entender o nível de acesso que a política concede. As ações da política são classificadas como `List`, `Read`, `Write`, `Permissions management` ou `Tagging`. Por exemplo, você pode escolher ações dos níveis de acesso `List` e `Read` para conceder acesso somente leitura a seus usuários. Para saber como usar resumos de políticas para entender as permissões no nível de acesso, consulte Níveis de acesso em resumos de política.
- Validar suas políticas: você pode executar a validação de políticas usando o IAM Access Analyzer ao criar e editar políticas de JSON. Recomendamos que você revise e valide todas as políticas existentes. O IAM Access Analyzer fornece mais de 100 verificações de política para validar suas

políticas. Ele gera avisos de segurança quando uma instrução em sua política permite o acesso que consideramos excessivamente permissivo. É possível usar as recomendações práticas fornecidas por meio dos avisos de segurança à medida que trabalha para conceder privilégios mínimos. Para saber mais sobre as verificações de política fornecidas pelo IAM Access Analyzer, consulte Validação da política do IAM Access Analyzer.

- Gerar uma política com base na atividade de acesso: para ajudar você a refinar as permissões que você concede, gere uma política do IAM baseada na atividade de acesso de uma entidade do IAM (usuário ou função). O IAM Access Analyzer revisa seus logs do AWS CloudTrail e gera um modelo de política que contém as permissões que foram usadas pela entidade no período especificado. É possível usar o modelo para criar uma política gerenciada com permissões refinadas e anexá-la à entidade do IAM. Dessa forma, você concede apenas as permissões de que o usuário ou a função precisa para interagir com os recursos da AWS para seu caso de uso específico. Para saber mais, consulte Geração de políticas do IAM Access Analyzer.
- Usar as informações do último acesso: outro recurso que pode ajudar com privilégios mínimos são as informações do último acesso. Visualize essas informações na guia Access Advisor (Consultor de acesso) na página de detalhes do console do IAM de um usuário, um grupo, uma função ou uma política do IAM. As informações do último acesso também incluem informações sobre as ações que foram acessadas pela última vez para alguns serviços, como Amazon EC2, IAM, Lambda e Amazon S3. Se você fizer login usando credenciais da conta de gerenciamento do AWS Organizations, poderá visualizar as informações do último acesso ao serviço na seção AWS Organizations do console do IAM. Você também pode usar a AWS CLI ou a API da AWS para recuperar um relatório das informações acessadas por último de entidades ou políticas no IAM ou no AWS Organizations. Você pode usar esta informação para identificar permissões desnecessárias, de forma que você possa refinar suas políticas do IAM ou do AWS Organizations para melhor aderir ao princípio de privilégio mínimo. Para obter mais informações, consulte Refinar permissões na AWS usando informações do último acesso.
- Revisar eventos da conta no AWS CloudTrail: para reduzir ainda mais as permissões, você pode visualizar os eventos da sua conta em AWS CloudTrail Event history (Histórico do evento). Os logs de eventos do CloudTrail incluem informações de eventos detalhadas que você pode usar para reduzir as permissões da política. Os logs incluem apenas as ações e os recursos de que suas entidades do IAM precisam. Para obter mais informações, consulte Visualizar eventos do CloudTrail no console do CloudTrail no Guia do usuário do AWS CloudTrail.

Para obter mais informações, consulte os tópicos de políticas para serviços individuais a seguir, os quais fornecem exemplos de como gravar políticas para recursos específicos do serviço.

- Usar políticas baseadas em recursos para o DynamoDB no Guia do desenvolvedor do Amazon DynamoDB
- Políticas de bucket para Amazon S3 no Guia do usuário do Amazon Simple Storage Service.
- Visão geral da lista de controle de acesso (ACL) no Guia do usuário do Amazon Simple Storage Service

Políticas gerenciadas e em linha

Ao definir as permissões para uma identidade no IAM, você deve decidir se deseja usar uma política gerenciada pela AWS, uma política gerenciada pelo cliente ou uma política em linha. Os tópicos a seguir oferecem mais informações sobre cada um dos tipos de políticas baseadas em identidade e quando usá-las.

A tabela a seguir descreve essas políticas:

Tipo de política	Descrição	Quem gerencia a política?	Modificar permissões?	Número de entidades principais aplicadas à política?
AWSPolíticas gerenciadas pela	Política autônoma criada e administrada pela AWS.	AWS	Não	Muitas
Políticas gerenciadas pelo cliente	Política que você cria para casos de uso específicos, e você poderá alterá-las ou atualizá-las com a frequência que desejar.	You	Sim	Muitas

Tipo de política	Descrição	Quem gerencia a política?	Modificar permissões?	Número de entidades principais aplicadas à política?
Políticas em linha	Política criada para uma única identidade do IAM (usuário, grupo ou perfil) que mantém uma estrita relação de um para um entre uma política e uma identidade.	You	Sim	Um

Para obter instruções sobre como criar perfis e anexar essas políticas, consulte Criar funções e anexar políticas (console).

Tópicos

- AWSPolíticas gerenciadas pela
- Políticas gerenciadas pelo cliente
- Políticas em linha
- Escolher entre políticas gerenciadas e políticas em linha
- Converter uma política em linha em uma política gerenciada
- Políticas gerenciadas pela AWS defasadas

AWSPolíticas gerenciadas pela

Uma política gerenciada pela AWS é uma política independente que é criada e administrada pela AWS. A política independente significa que a política tem seu próprio nome do recurso da Amazon (ARN) que inclui o nome da política. Por exemplo, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` é uma política gerenciada da AWS. Para obter mais informações sobre ARNs

do , consulte ARNs do IAM. Para obter uma lista das políticas gerenciadas pela AWS dos Serviços da AWS, consulte Políticas gerenciadas pela AWS.

AWSAs políticas gerenciadas pela facilitam a atribuição das devidas permissões aos usuários, perfis e grupos do IAM. É mais rápido do que escrever as políticas por conta própria e contém permissões para vários casos de uso comuns.

Você não pode alterar as permissões definidas em políticas gerenciadas pela AWS. Ocasionalmente, a AWS atualizará as permissões definidas em uma política gerenciada da AWS. Quando a AWS faz isso, a atualização afeta todas as entidades principais (usuários, grupos e perfis do IAM) às quais a política está anexada. É mais provável que a AWS atualize uma política gerenciada pela AWS quando um novo serviço da AWS for iniciado ou novas chamadas de API se tornarem disponíveis para os serviços existentes. Por exemplo, a política gerenciada pela AWS denominada ReadOnlyAccess fornece acesso somente leitura a todos os recursos e Serviços da AWS. Quando a AWS lança um novo serviço, a AWS atualiza a política ReadOnlyAccess para adicionar permissões somente leitura para o novo serviço. As permissões atualizadas são aplicadas a todas as entidades principais às quais política estiver anexada.

Políticas gerenciadas pela AWS de acesso total: elas definem permissões para administradores de serviço concedendo acesso total a um serviço. Os exemplos incluem:

- AmazonDynamoDBFullAccess
- IAMFullAccess

Políticas gerenciadas pela AWS para usuários avançados: essas políticas fornecem acesso total aos recursos e Serviços da AWS, mas não permitem gerenciar usuários e grupos do IAM. Os exemplos incluem:

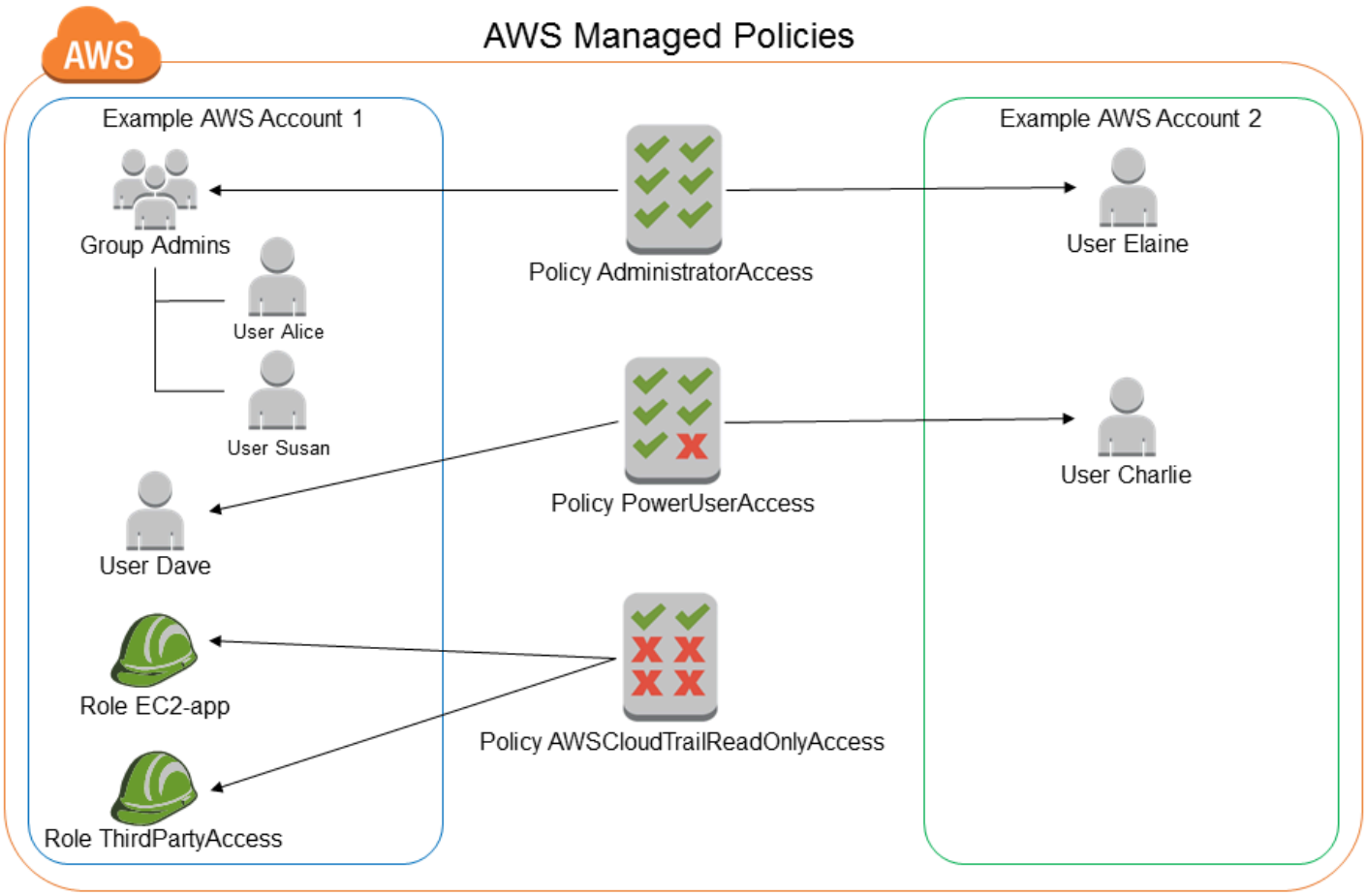
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

Políticas gerenciadas pela AWS de acesso parcial: elas fornecem níveis específicos de acesso aos Serviços da AWS sem fornecer permissões de nível de acesso de gerenciamento de permissões. Os exemplos incluem:

- AmazonMobileAnalyticsWriteOnlyAccess
- AmazonEC2ReadOnlyAccess

Políticas gerenciadas pela AWS de funções de trabalho: essas políticas se alinham estreitamente a funções de trabalho bastante usadas no setor de TI e facilitam a concessão de permissões para essas funções de trabalho. Uma das principais vantagens de usar políticas de função de trabalho é que elas são mantidas e atualizadas pela AWS à medida que novos serviços e operações de API são introduzidos. Por exemplo, a função de trabalho AdministratorAccess fornece acesso total e delegação de permissões para cada serviço e recurso na AWS. Recomendamos usar essa política apenas para o administrador da conta. Para usuários avançados que exigem acesso completo a todos os serviços, exceto o acesso limitado ao IAM e ao AWS Organizations, use a função de trabalho PowerUserAccess. Para obter uma lista e as descrições das políticas de função de trabalho, consulte AWSPolíticas gerenciadas pela para funções de trabalho.

O seguinte diagrama ilustra as políticas gerenciadas pela AWS. O diagrama mostra três políticas gerenciadas pela AWS: AdministratorAccess, PowerUserAccess e AWSCloudTrailReadOnlyAccess. Uma única política gerenciada pela AWS pode ser anexada a entidades principais em diferentes Contas da AWS e a entidades principais diferentes em uma única Conta da AWS.



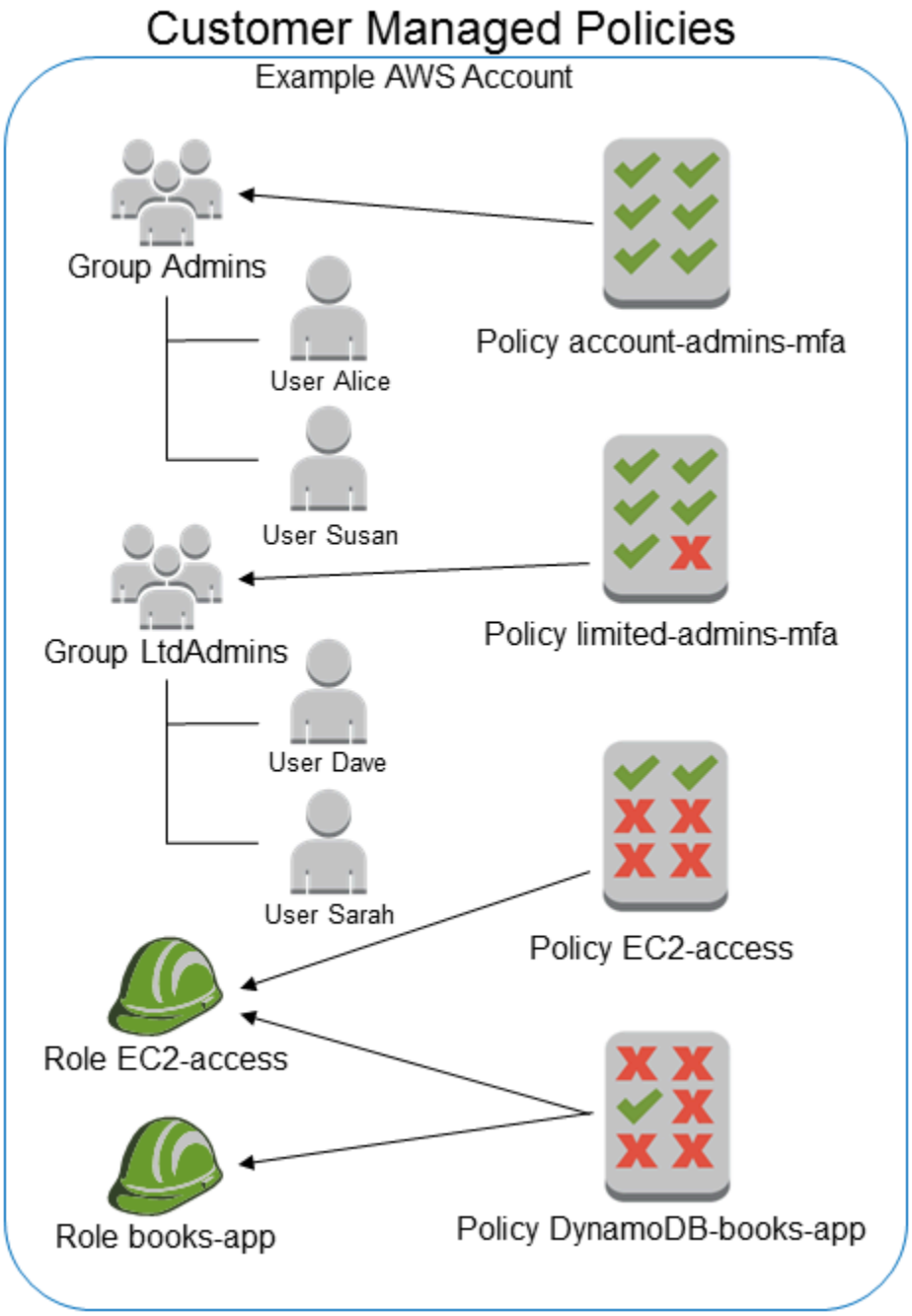
Políticas gerenciadas pelo cliente

Você pode criar políticas autônomas na sua Conta da AWS, que podem ser anexadas às entidades principais (usuários, grupos e perfis do IAM). Crie essas políticas gerenciadas pelo cliente para seus casos de uso específicos e você poderá alterá-las e atualizá-las quantas vezes quiser. Assim como nas políticas gerenciadas pela AWS, ao anexar uma política a uma entidade principal, você atribui à entidade as permissões que estão definidas na política. Quando você atualiza permissões na política, as alterações são aplicadas a todas as entidades principais às quais a política esteja anexada.

Uma ótima forma de criar uma política gerenciada pelo cliente é começar copiando uma política gerenciada pela AWS. Dessa forma, você sabe que a política está correta no início e basta personalizá-la para seu ambiente.

O seguinte diagrama ilustra as políticas gerenciadas pelo cliente. Cada política é uma entidade no IAM com seu próprio nome de recurso da Amazon (ARN) que inclui o nome da política. Observe que a mesma política pode ser anexada a várias entidades de segurança, por exemplo, a mesma política DynamoDB-books-app é anexada a duas funções do IAM distintas.

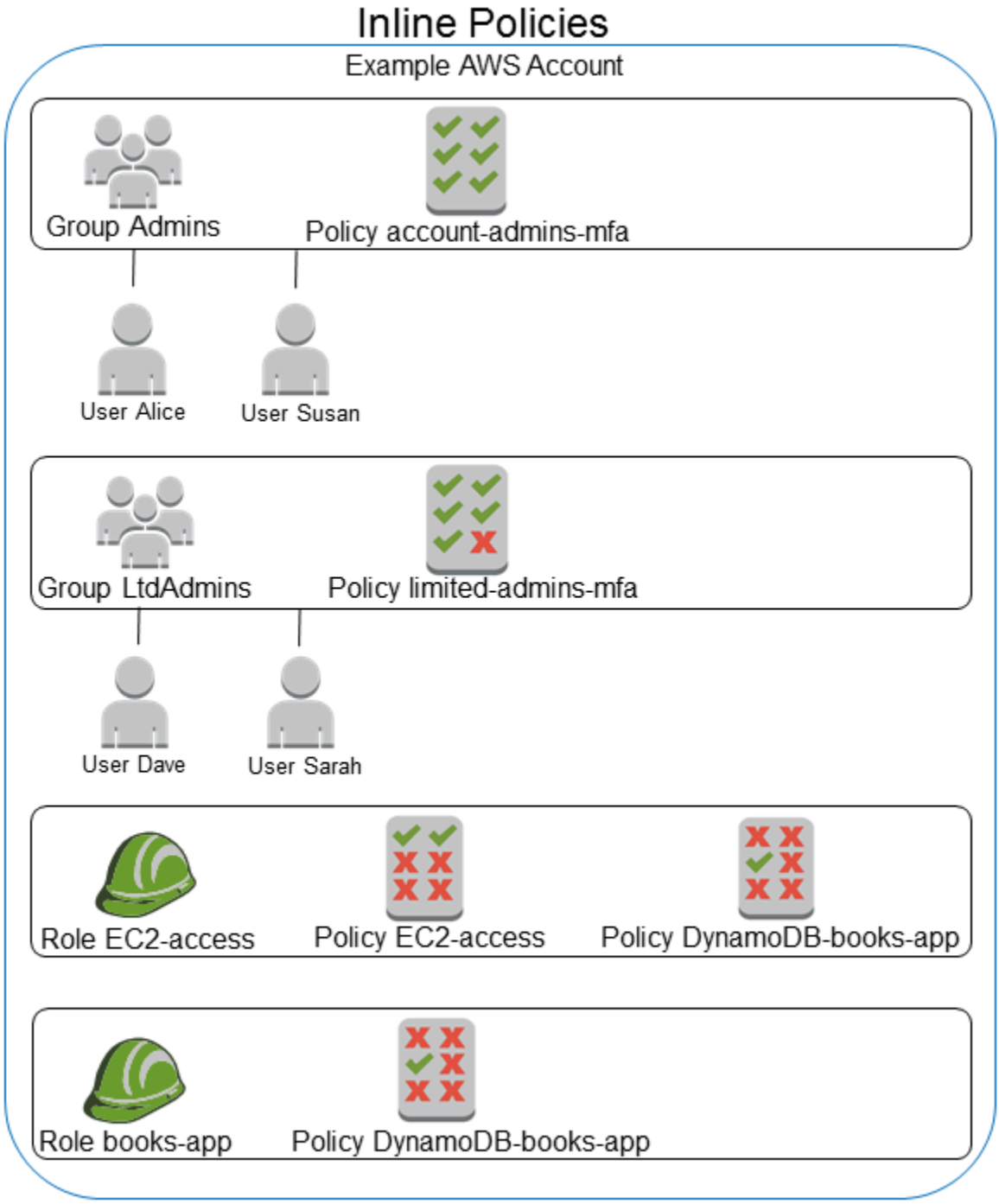
Para obter mais informações, consulte . Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente



Políticas em linha

A política em linha é uma política criada para uma única identidade do IAM (um usuário, grupo de usuários ou perfil). As políticas em linha mantêm um relacionamento estrito de um para um entre uma política e uma identidade. Elas são excluídas quando você exclui a identidade. Você pode criar uma política e incorporá-la em uma identidade, seja ao criar a identidade ou posteriormente. Se a política puder ser aplicada a mais de uma entidade, é melhor usar uma política gerenciada.

O seguinte diagrama ilustra as políticas em linha. Cada política é uma parte inerente do usuário, do grupo ou da função. Observe que dois perfis incluem a mesma política (DynamoDB-books-app), mas eles não compartilham uma única política. Cada perfil tem sua própria cópia da política.



Escolher entre políticas gerenciadas e políticas em linha

Considere seus casos de uso ao decidir entre políticas gerenciadas e em linha. Na maioria dos casos, recomendamos que você use políticas gerenciadas em vez de políticas em linha.

> **Note**
>
> É possível usar políticas gerenciadas e em linha juntas para definir permissões comuns e exclusivas para uma entidade principal.

As políticas gerenciadas fornecem os seguintes recursos:

Capacidade de reutilização

Uma única política gerenciada pode ser anexada a várias entidades principais (usuários, grupos e funções). Você pode criar uma biblioteca de políticas que definem permissões úteis para sua Conta da AWS e anexar essas políticas a entidades principais, conforme necessário.

Gerenciamento de alterações central

Quando você alterar uma política gerenciada, a alteração será aplicada a todas as entidades principais às quais a política estiver anexada. Por exemplo, se você quiser adicionar permissões para uma nova API da AWS, poderá atualizar uma política gerenciada pelo cliente ou associar uma política gerenciada pela AWS para adicionar a permissão. Se você estiver usando uma política gerenciada pela AWS, a AWS atualizará a política. Quando uma política gerenciada é atualizada, as alterações são aplicadas a todas as entidades principais às quais a política gerenciada está anexada. Por outro lado, para alterar uma política em linha, é necessário editar individualmente cada identidade que a contém. Por exemplo, se um grupo e um perfil contiverem a mesma política em linha, você deverá editar individualmente as duas entidades principais para alterar essa política.

Versionamento e reversão

Quando você altera uma política gerenciada pelo cliente, a política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. O IAM armazena até cinco versões de suas políticas gerenciadas pelo cliente. Você pode usar as versões da política para reverter uma política para uma versão anterior, conforme necessário.

> **Note**
>
> Uma versão de política é diferente de um elemento de política `Version`. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. Para saber mais sobre as versões de política, consulte the section called “Versionamento de políticas do IAM”. Para saber mais sobre o elemento de política `Version`, consulte Elementos de política JSON do IAM: Version.

Como delegar o gerenciamento de permissões

Você pode permitir que os usuários na sua Conta da AWS anexem e desanexem políticas sem deixar de manter o controle das permissões definidas nessas políticas. Para isso, você pode designar alguns usuários como administradores completos, ou seja, administradores que podem criar, atualizar e excluir políticas. Em seguida, você pode designar outros usuários como administradores limitados. Esses administradores limitados que podem anexar políticas a outras entidades principais, mas somente as políticas que você permitiu que eles anexassem.

Para obter mais informações sobre como delegar permissões, consulte Controle de acesso a políticas.

Limites de caracteres de política maiores

O limite máximo de tamanho em caracteres para as políticas gerenciadas é maior do que o limite para as políticas em linha. Se você atingir o limite de tamanho em caracteres da política em linha, poderá criar mais grupos do IAM e anexar a política gerenciada ao grupo.

Para obter mais informações sobre cotas e limites, consulte IAM e cotas do AWS STS.

Atualizações automáticas para políticas gerenciadas pela AWS

A AWS mantém políticas gerenciadas pela AWS e as atualiza quando necessário, por exemplo, para adicionar permissões para novos serviços da AWS), sem precisar fazer alterações. As atualizações são aplicadas automaticamente às entidades principais às quais você tenha anexado a política gerenciada pela AWS.

Introdução a serviços gerenciados

Recomendamos o uso de políticas que concedam privilégios mínimos ou conceder apenas as permissões necessárias para executar uma tarefa. A maneira mais segura de conceder privilégio

mínimo é escrever uma política gerenciada pelo cliente apenas com as permissões necessárias para sua equipe. Você deve criar um processo para permitir que sua equipe solicite mais permissões quando necessário. É necessário tempo e experiência para criar políticas gerenciadas pelo cliente do IAM que fornecem à sua equipe apenas as permissões de que precisam.

Para começar a adicionar permissões às suas identidades do IAM (usuários, grupos de usuários e funções), você pode usar as AWSPolíticas gerenciadas pela. As políticas gerenciadas pela AWS não concedem permissões de privilégio mínimo. Você deve considerar o risco de segurança de conceder às suas entidades de segurança mais permissões do que elas precisam para realizar um trabalho.

Você pode anexar políticas gerenciadas pela AWS, incluindo funções de trabalho, a qualquer identidade do IAM. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

Para alternar para permissões de privilégio mínimo, você pode executar o AWS Identity and Access Management and Access Analyzer para monitorar as entidades de segurança com políticas gerenciadas pela AWS. Depois de saber quais permissões elas estão usando, você pode escrever ou gerar uma política gerenciada pelo cliente apenas com as permissões necessárias para sua equipe. Isso é menos seguro, mas oferece mais flexibilidade à medida que você aprende como sua equipe está usando a AWS. Para obter mais informações, consulte Geração de política do IAM Access Analyzer.

AWSAs políticas gerenciadas pela são criadas para fornecer permissões para vários casos de uso comuns. Para obter mais informações sobre políticas gerenciadas pela AWS projetadas para funções de trabalho específicas, consulte AWSPolíticas gerenciadas pela para funções de trabalho.

Para obter uma lista de políticas gerenciadas pela AWS, consulte o Guia de referência de políticas gerenciadas pela AWS.

Usar políticas em linha

As políticas em linha são úteis se você desejar manter um relacionamento rigorosamente individual entre uma política e a identidade à qual ela é aplicada. Por exemplo, se você deseja ter certeza de que as permissões em uma política não sejam atribuídas acidentalmente a uma identidade diferente da pretendida. Quando você usa uma política em linha, as permissões nela não podem ser anexadas acidentalmente à identidade errada. Além disso, quando você usa o Console de gerenciamento da AWS para excluir a identidade, as políticas incorporadas nela também são excluídas porque fazem parte da entidade principal.

Converter uma política em linha em uma política gerenciada

Se tiver políticas em linha na conta, você poderá convertê-las em políticas gerenciadas. Para fazer isso, copie a política para uma nova política gerenciada. Depois, anexe a nova política à identidade que tem a política em linha. Depois disso, exclua a política em linha.

Converter uma política em linha em uma política gerenciada

Para converter uma política em linha em uma política gerenciada

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usários), Users (Usuários) ou Roles (Funções).
3. Na lista, selecione o nome do grupo de usuários, usuário ou função que tem a política que você deseja remover.
4. Escolha a aba Permissões.
5. Para grupos do IAM, selecione o nome da política em linha que você deseja remover. Em usuários e perfis, escolha Mostrar mais ***n***, se necessário, e depois expanda a política em linha que você deseja remover.
6. Escolha Copiar para copiar o documento da política JSON para a política.
7. No painel de navegação, selecione Políticas.
8. Escolha Criar política e depois escolha a opção JSON.
9. Substitua o texto existente pelo texto da sua política JSON e escolha Avançar.
10. Insira um nome e uma descrição opcional para a política e escolha Criar política.
11. No painel de navegação, escolha User groups (Grupos de usuários), Users (Usuários) ou Roles (Funções) e, novamente, escolha o nome do grupo de usuários, usuário ou função que tem a política que você deseja remover.
12. Escolha a guia Permissões e depois escolha Adicionar permissões.
13. Para grupos do IAM, marque a caixa de seleção ao lado do nome da nova política, escolha Adicionar permissões e depois Anexar política. Para usuários ou funções, escolha Add permissions (Adicionar permissões). Na página seguinte, escolha Anexar políticas existentes diretamente, marque a caixa de seleção ao lado do nome da nova política, escolha Próximo e depois Adicionar permissões.

Você será direcionado para a página Summary (Resumo) do grupo de usuários, do usuário ou da função.

14. Marque a caixa de seleção ao lado da política em linha que você deseja remover e escolha Remover.

Políticas gerenciadas pela AWS defasadas

Para simplificar a atribuição de permissões, a AWS fornece políticas gerenciadas, políticas predefinidas que estão prontas para serem anexadas a usuários, grupos e funções do IAM.

Às vezes, a AWS precisa adicionar uma nova permissão a uma política existente, como quando um novo serviço é introduzido. Adicionar uma nova permissão a uma política existente não interrompe nem remove nenhum recurso ou capacidade.

No entanto, a AWS pode optar por criar uma nova política quando as alterações necessárias podem afetar os clientes se elas forem aplicadas a uma política existente. Por exemplo, a remoção de permissões de uma política existente pode interromper as permissões de alguma entidade do IAM ou aplicação que depende dela, podendo interromper uma operação essencial.

Portanto, quando uma alteração é necessária, a AWS cria uma nova política inteiramente nova com as alterações necessárias e a disponibiliza para os clientes. A política antiga é, então, marcada como preterida. Para obter mais informações, consulte Deprecated AWS managed policies no AWS Managed Policy Reference Guide.

Estabeleça barreiras de proteção para permissões usando perímetros de dados

As barreiras de proteção do perímetro de dados devem servir como limites sempre ativos para ajudar a proteger seus dados em um amplo conjunto de contas e recursos da AWS. Os perímetros de dados seguem as práticas recomendadas de segurança do IAM para estabelecer barreiras de proteção de permissões em várias contas. Essas barreiras de proteção de permissões em toda a organização não substituem seus controles de acesso refinados existentes. Em vez disso, elas funcionam como controles de acesso de baixa granularidade que ajudam a melhorar sua estratégia de segurança, garantindo que usuários, perfis e recursos sigam um conjunto de padrões de segurança definidos.

Um perímetro de dados é um conjunto de barreiras de proteção de permissão em seu ambiente da AWS que ajudam a garantir que somente suas identidades confiáveis acessem recursos confiáveis das redes esperadas.

- Identidades confiáveis: entidades principais (perfis ou usuários do IAM) em suas AWS contas e AWS serviços agindo em seu nome.
- Recursos confiáveis: recursos de propriedade de suas contas da AWS ou de serviços da AWS que atuam em seu nome.
- Redes esperadas: seus data centers on-premises e nuvens privadas virtuais (VPCs) ou redes de serviços da AWS agindo em seu nome.

> **Note**
>
> Em alguns casos, talvez seja necessário ampliar o perímetro de dados para incluir o acesso de seus parceiros comerciais confiáveis. Você deve considerar todos os padrões de acesso aos dados pretendidos ao criar uma definição de identidades e recursos confiáveis e redes esperadas específicas para sua empresa e seu uso dos Serviços da AWS.

Os controles de perímetro de dados devem ser tratados como qualquer outro controle de segurança dentro do programa de gerenciamento de riscos e segurança da informação. Isso significa que você deve realizar uma análise de ameaças para identificar potenciais riscos em seu ambiente de nuvem e, com base em seus próprios critérios de aceitação de riscos, selecionar e implementar controles de perímetro de dados apropriados. Para melhor informar a abordagem iterativa baseada em riscos para a implementação do perímetro de dados, você precisa entender quais riscos de segurança e vetores de ameaças são abordados pelos controles de perímetro de dados, bem como suas prioridades de segurança.

Controles de perímetro de dados

Os controles de baixa granularidade do perímetro de dados ajudam você a atingir seis objetivos de segurança distintos em três perímetros de dados por meio da implementação de diferentes combinações de Tipos de políticas e chaves de condição.

Perímetro	Objetivo de controle	Utilizar	Aplicado em	Chaves de contexto de condições globais
Identidade	Somente identidades confiáveis podem acessar meus recursos	RCP	Recursos	aws:PrincipalOrgID aws:PrincipalOrgPaths
	Somente identidades confiáveis são permitidas na minha rede	Política de endpoint da VPC	Rede	aws:PrincipalAccount aws:PrincipalIsAwsService aws:SourceOrgID aws:SourceOrgPath aws:SourceAccount
Recursos	Suas identidades podem acessar apenas recursos confiáveis	SCP	Identidades	aws:ResourceOrgID aws:ResourceOrgPaths
	Somente recursos confiáveis podem ser acessados de sua rede	Política de endpoint da VPC	Rede	aws:ResourceAccount

Perímetro	Objetivo de controle	Utilizar	Aplicado em	Chaves de contexto de condições globais
Rede	Suas identidades podem acessar recursos somente das redes esperadas	SCP	Identidades	aws:SourceIp aws:SourceVpc aws:SourceVpce aws:VpceAccount aws:VpceOrgPaths aws:VpceOrgID aws:ViaAWSService aws:PrincipalIsAwsService
	Seus recursos somente podem acessados de redes esperadas	RCP	Recursos	

É possível pensar nos perímetros de dados como a criação de um limite firme em torno de seus dados para evitar padrões de acesso não intencionais. Embora os perímetros de dados possam impedir um amplo acesso não intencional, você ainda precisa tomar decisões de controle de acesso refinadas. Estabelecer um perímetro de dados não diminui a necessidade de ajustar continuamente as permissões usando ferramentas como o IAM Access Analyzer como parte de sua jornada para obter o privilégio mínimo.

Para impor controles de perímetro de dados em recursos para os quais as RPCs não oferecem suporte no momento, é possível usar políticas baseadas em recursos que são anexadas diretamente aos recursos. Para obter uma lista de serviços que oferecem suporte a RCPs e a políticas baseadas em recursos, consulte Políticas de controle de recursos (RCPs) e AWSServiços da que funcionam com o IAM.

Para impor controles de perímetro de rede, recomendamos que você use `aws:VpceOrgID`, `aws:VpceOrgPaths` e `aws:VpceAccount` somente se todos os serviços aos quais você deseja restringir o acesso forem atualmente suportados. O uso dessas chaves de condição com serviços não suportados pode levar a resultados de autorização não intencionais. Para obter uma lista de serviços compatíveis as chaves, consulte as AWSChaves de contexto de condições globais da. Se você precisar aplicar os controles em uma variedade maior de serviços, considere usar `aws:SourceVpc` e `aws:SourceVpce` em vez disso.

Perímetro de identidade

Um perímetro de identidade é um conjunto de controles de acesso preventivos de baixa granularidade que ajudam a garantir que somente identidades confiáveis possam acessar seus recursos e que somente identidades confiáveis sejam permitidas em sua rede. As identidades confiáveis geralmente incluem entidades principais (perfis ou usuários) em suas contas da AWS e serviços da AWS que atuam em seu nome. Todas as outras identidades são consideradas não confiáveis e são impedidas pelo perímetro de identidade, a menos que uma exceção explícita seja concedida.

As seguintes chaves de condição globais ajudam a impor controles de perímetro de identidade com base na sua definição de identidades confiáveis. Use essas chaves nas políticas de controle de recursos para restringir o acesso a recursos, ou nas políticas de endpoint da VPC para restringir o acesso a suas redes.

Identidades pertencentes a você

Você pode usar as seguintes chaves de condição para definir as entidades principais do IAM que você cria e gerencia em sua Contas da AWS.

- aws:PrincipalOrgID — É possível usar essa chave de condição para garantir que as entidades principais do IAM que fazem a solicitação pertençam à organização especificada em AWS Organizations.
- aws:PrincipalOrgPaths — É possível usar essa chave de condição para certificar-se de que o usuário do IAM, o perfil do IAM, a entidade principal do usuário federado do AWS STS, a entidade principal federada SAML, a entidade principal federada OIDC ou a Usuário raiz da conta da AWS que fez a solicitação pertençam à unidade organizacional (UO) especificada no AWS Organizations.
- aws:PrincipalAccount — É possível usar essa chave de condição para garantir que os recursos só possam ser acessados pela conta da entidade principal especificada na política.

Identidades dos serviços da AWS que atuam em seu nome

Você pode usar as seguintes chaves de condição para permitir que os serviços da AWS usem suas próprias identidades para acessar seus recursos quando agirem em seu nome.

- aws:PrincipalIsAWSService e aws:SourceOrgID (ou aws:SourceOrgPaths e aws:SourceAccount) — É possível usar essas chaves de condição para garantir que, quando as entidades principais do AWS service (Serviço da AWS) acessarem seus recursos, eles o façam somente em nome de um recurso na organização, unidade organizacional ou conta especificada em AWS Organizations.

Para obter mais informações, consulte Estabelecendo um perímetro de dados em AWS: permitir que somente identidades confiáveis acessem os dados da empresa.

Perímetro de recurso

Um perímetro de recurso é um conjunto de controles de acesso preventivos de baixa granularidade que ajudam a garantir que suas identidades possam acessar somente recursos confiáveis e que somente recursos confiáveis possam ser acessados de sua rede. Os recursos confiáveis geralmente incluem recursos de propriedade de suas contas da AWS ou de serviços da AWS que atuam em seu nome.

As seguintes chaves de condição globais ajudam a impor controles de perímetro de recursos com base na sua definição de recursos confiáveis. Use essas chaves nas Políticas de controle de serviços (SCPs) para restringir quais recursos podem ser acessados por suas identidades ou nas Políticas de endpoint da VPC para restringir quais recursos podem ser acessados de suas redes.

Recursos pertencentes a você

Você pode usar as seguintes chaves de condição para definir os recursos da AWS que você cria e gerencia em suas Contas da AWS.

- aws:ResourceOrgID — É possível usar essa chave de condição para garantir que o recurso que está sendo acessado pertence à organização especificada em AWS Organizations.
- aws:ResourceOrgPaths — É possível usar essa chave de condição para garantir que o recurso que está sendo acessado pertence à unidade organizacional (UO) especificada em AWS Organizations.
- aws:ResourceAccount – É possível usar essa chave de condição para garantir que o recurso que está sendo acessado pertence à Conta da AWS especificada.

Recursos dos serviços da AWS que atuam em seu nome

Em alguns casos, talvez seja necessário permitir o acesso a recursos próprios da AWS, recursos que não pertencem à sua organização e que são acessados por suas entidades principais ou por serviços da AWS que atuam em seu nome. Para obter mais informações sobre esses cenários, consulte Estabelecendo um perímetro de dados emAWS: permitir somente recursos confiáveis da minha organização.

Perímetro de rede

Um perímetro de rede é um conjunto de controles de acesso preventivos de baixa granularidade que ajudam a garantir que suas identidades possam acessar recursos somente de redes esperadas e que seus recursos só possam ser acessados de redes esperadas. As redes esperadas geralmente incluem seus data centers on-premises e nuvens privadas virtuais (VPCs) e redes de serviços da AWS agindo em seu nome.

As seguintes chaves de condição globais ajudam a impor controles de perímetro de rede com base na sua definição de redes esperadas. Use essas chaves nas políticas de controle de serviços (SCPs) para restringir as redes com as quais suas identidades podem se comunicar, ou nas políticas de controle de recursos (RCPs) para restringir o acesso aos recursos às redes esperadas.

Redes pertencentes a você

Você pode usar as seguintes chaves de condição para definir as redes que seus funcionários e aplicações devem usar para acessar seus recursos, como seu intervalo de IP CIDR corporativo e suas VPCs.

- aws:SourceIp – É possível usar essa chave de condição para garantir que o endereço IP do solicitante esteja dentro de um intervalo de IP especificado.
- aws:SourceVpc – É possível usar essa chave de condição para garantir que o endpoint da VPC pelo qual a solicitação passa pertença à VPC especificada.
- aws:SourceVpce – É possível usar essa chave de condição para garantir que a solicitação passe pelo endpoint da VPC especificado.
- aws:VpceAccount – É possível usar essa chave de condição para garantir que as solicitações cheguem por meio de endpoints da VPC pertencentes à conta da AWS especificada.
- aws:VpceOrgPaths – É possível usar essa chave de condição para garantir que as solicitações cheguem por meio de endpoints da VPC pertencentes a contas na unidade organizacional (UO) especificada em AWS Organizations.

- aws:VpceOrgID – É possível usar essa chave de condição para garantir que as solicitações cheguem por meio de endpoints da VPC pertencentes a contas na organização especificada em AWS Organizations.

`aws:VpceAccount`, `aws:VpceOrgPaths`, e `aws:VpceOrgID` são particularmente úteis para implementar controles de perímetro de rede que escalam automaticamente com o uso do seu endpoint da VPC, sem exigir atualizações nas políticas ao criar novos endpoints. Consulte a AWSChaves de contexto de condições globais da para obter a lista de Serviços da AWS compatíveis essas chaves.

Redes dos serviços da AWS que atuam em seu nome

Você pode usar as seguintes chaves de condição para permitir que os serviços da AWS acessem seus recursos a partir de suas redes quando agirem em seu nome.

- aws:ViaAWSService – É possível usar essa chave de condição para garantir que Serviços da AWS possa fazer solicitações em nome de sua entidade principal usando Sessões de acesso direto (FAS).
- aws:PrincipalIsAWSService – É possível usar essa chave de condição para garantir que Serviços da AWS possa acessar seus recursos usando AWSResponsáveis pelos serviços da.

Há cenários adicionais em que você precisa permitir o acesso aos Serviços da AWS que acessa esses recursos de fora da rede. Para obter mais informações, consulte Estabelecendo um perímetro de dados em AWS: permitir acesso aos dados da empresa somente de redes confiáveis.

Recursos para saber mais sobre perímetros de dados

Os seguintes recursos podem ajudá-lo a saber mais sobre os perímetros de dados em todo a AWS.

- Perímetros de dadosna AWS — Saiba mais sobre os perímetros de dados e seus benefícios e casos de uso.
- Série de postagens do blog: Estabelecendo um perímetro de dados na AWS — Essas postagens do blog abordam orientações prescritivas sobre como estabelecer seu perímetro de dados em grande escala, incluindo considerações importantes sobre segurança e implementação.
- Exemplos de políticas de perímetro de dados — Este repositório do GitHub contém exemplos de políticas que abrangem alguns padrões comuns para ajudar você a implementar um perímetro de dados na AWS.

- Auxiliar de perímetro de dados — Essa ferramenta ajuda você a projetar e antecipar o impacto de seus controles de perímetro de dados analisando a atividade de acesso em seus logs AWS CloudTrail.

Limites de permissões para entidades do IAM

A AWS oferece suporte a limites de permissões para entidades (usuários ou funções) do IAM. Um limite de permissões é um recurso avançado para usar uma política gerenciada para definir as permissões máximas que uma política baseada em identidade pode conceder a uma entidade do IAM. O limite de permissões de uma entidade permite que a entidade execute somente as ações permitidas por ambas as políticas baseadas em identidade e seus limites de permissões.

Para obter mais informações sobre tipos de política, consulte Tipos de políticas.

> **Important**
>
> Não use declarações de política baseadas em recursos que incluam um elemento de política `NotPrincipal` com um efeito `Deny` para usuários ou perfis do IAM que tenham uma política de limite de permissões anexada. O elemento `NotPrincipal` com um efeito `Deny` sempre negará qualquer entidade principal do IAM que tenha uma política de limite de permissões anexada, independentemente dos valores especificados no elemento `NotPrincipal`. Isso faz com que alguns usuários ou perfis do IAM que, de outra forma, teriam acesso ao recurso, percam o acesso.
> Recomendamos alterar suas instruções de política baseadas em recursos para usar o operador de condição `ArnNotEquals` com a chave de contexto `aws:PrincipalArn` para limitar o acesso, em vez do elemento `NotPrincipal`. Para obter informações sobre o elemento `NotPrincipal`, consulte Elementos da política JSON da AWS: NotPrincipal.

Você pode usar uma política gerenciada pela AWS ou uma política gerenciada pelo cliente para definir o limite para uma entidade (usuário ou função) do IAM. Essa política limita o número máximo de permissões para o usuário ou a função.

Por exemplo, suponha que o usuário do IAM chamado `Shirley` deva ter permissão para gerenciar apenas o Amazon S3, o Amazon CloudWatch e o Amazon EC2. Para impor essa regra, você pode usar a seguinte política para definir o limite de permissões para a usuária `Shirley`:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Quando você usa uma política para definir o limite de permissões para um usuário, ela limita as permissões do usuário, mas não fornece permissões por conta própria. Neste exemplo, a política define as permissões máximas de `Shirley` como todas as operações no Amazon S3, CloudWatch e Amazon EC2. Shirley nunca poderá executar operações em qualquer outro serviço, incluindo o IAM, mesmo que ela tenha uma política de permissões que permita isso. Por exemplo, você pode adicionar a política a seguir à usuária `Shirley`:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Esta política permite criar um usuário no IAM. Se você anexar essa política de permissões à usuária `Shirley`, e Shirley tentar criar um usuário, a operação falhará. A falha ocorre porque o limite de permissões não permite a operação `iam:CreateUser`. Dadas essas duas políticas, Shirley não

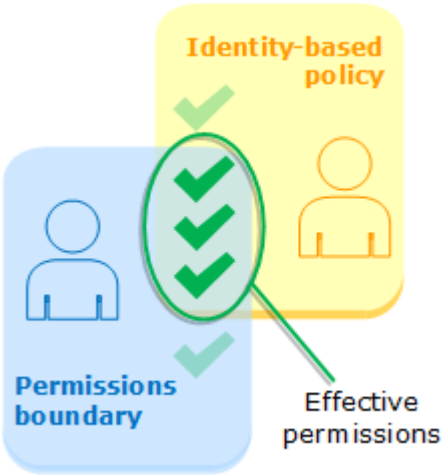
tem permissão para executar nenhuma operação na AWS. Você deve adicionar uma política de permissões diferente para permitir ações em outros serviços, como o Amazon S3. Como alternativa, você pode atualizar o limite de permissões para permitir que ela crie um usuário no IAM.

Avaliar permissões efetivas com limites

O limite de permissões para uma entidade do IAM (usuário ou função) define o número máximo de permissões que a entidade pode ter. Isso pode alterar as permissões efetivas para esse usuário ou função. As permissões efetivas para uma entidade são as permissões que são concedidas por todas as políticas que afetam o usuário ou a função. Dentro de uma conta, as permissões para uma entidade podem ser afetadas por políticas baseadas em identidade, políticas baseadas em recurso, limites de permissões, SCPs de AWS Organizations ou políticas de sessão. Para obter mais informações sobre os diversos tipos diferentes de políticas, consulte Políticas e permissões no AWS Identity and Access Management.

Se algum desses tipos de política negar explicitamente o acesso de uma operação, a solicitação será negada. As permissões concedidas a uma entidade por vários tipos de permissões são mais complexas. Para obter mais detalhes sobre como o AWS avalia as políticas, consulte Lógica da avaliação de política.

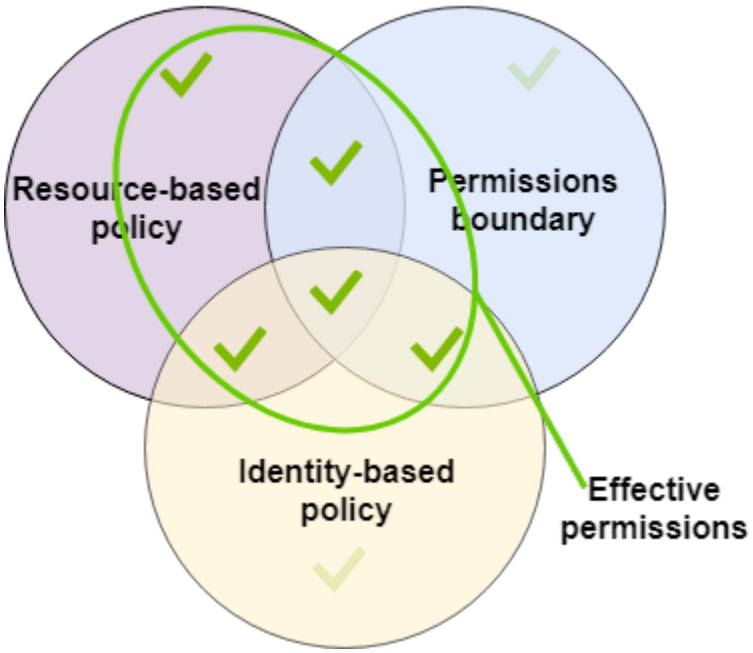
Políticas baseadas em identidade com limites: as políticas baseadas em identidade são políticas em linha ou gerenciadas que são anexadas a um usuário, grupo de usuários ou função. As políticas baseadas em identidade concedem permissão para a entidade, e os limites de permissões limitam essas permissões. As permissões efetivas são a interseção de ambos os tipos de política. Uma negação explícita em qualquer uma dessas políticas substitui a permissão.



Políticas baseadas em recurso: as políticas baseadas em recurso controlam como a entidade de segurança pode acessar o recurso ao qual a política está anexada.

Políticas baseadas em recursos para usuários do IAM

Na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de usuário do IAM (que não é uma sessão de entidade principal de usuário federado do AWS STS) não são limitadas por uma negação implícita em uma política baseada em identidade ou limite de permissões.



Políticas baseadas em recursos para funções do IAM

Função do IAM: as políticas baseadas em recursos que concedem permissões a um ARN de função do IAM são limitadas por uma negação implícita em um limite de permissões ou política de sessão.

Sessão de função do IAM: na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de sessão de função do IAM concedem permissões diretamente à sessão de função assumida. As permissões concedidas diretamente a uma sessão não são limitadas por uma negação implícita em uma política baseada em identidade, um limite de permissões ou uma política de sessão. Quando você assume uma função e faz uma solicitação, a entidade principal que faz a solicitação é o ARN da sessão de função do IAM e não o ARN da função em si.

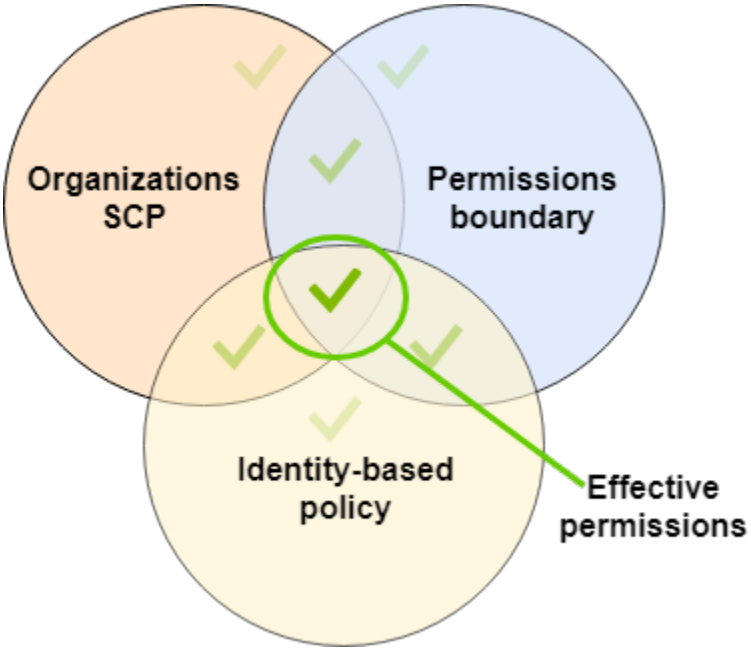
Políticas baseadas em recursos para sessões de entidades principais de usuários federados do AWS STS

Sessões de entidades principais de usuários federados do AWS STS: uma sessão de entidade principal de usuário federado do AWS STS é uma sessão criada mediante o chamado de

`GetFederationToken`. Quando um usuário federado faz uma solicitação, a entidade principal que faz a solicitação é o ARN do usuário federado e não o ARN do usuário do IAM que federou. Na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de usuário federado concedem permissões diretamente para a sessão. As permissões concedidas diretamente a uma sessão não são limitadas por uma negação implícita em uma política baseada em identidade, um limite de permissões ou uma política de sessão.

No entanto, se uma política baseada em recursos conceder permissão ao ARN do usuário do IAM que se federou, as solicitações feitas pela entidade principal do usuário federado do AWS STS durante a sessão serão limitadas por uma negação implícita em um limite de permissão ou política de sessão.

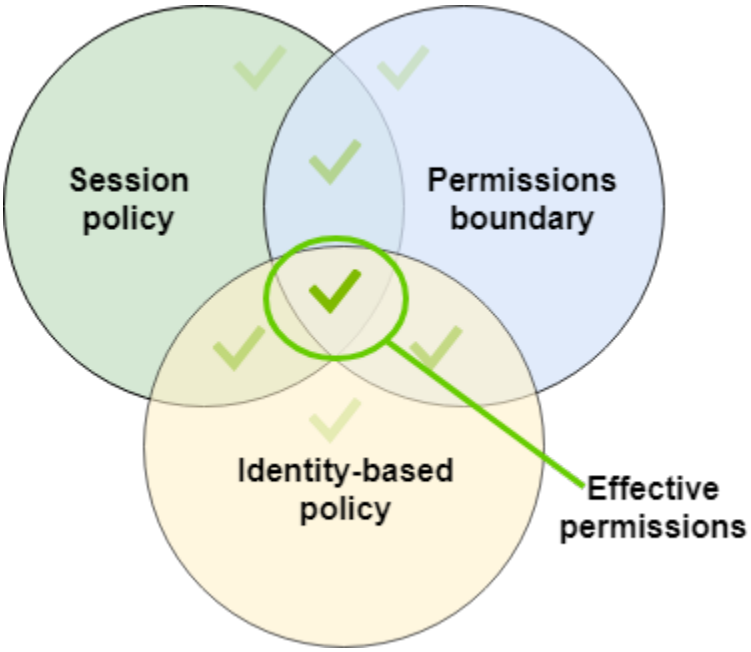
SCPs do AWS Organizations: as SCPs são aplicadas a uma conta inteira do Conta da AWS. Eles limitam as permissões para todas as solicitações feitas por um principal na conta. Uma entidade (usuário ou função) do IAM pode fazer uma solicitação que é afetada por uma SCP, um limite de permissões e uma política baseada em identidade. Nesse caso, a solicitação é permitida somente se todos os três tipos de política a permitem. As permissões efetivas são a interseção de todos os três tipos de política. Uma negação explícita em qualquer uma dessas políticas substitui a permissão.



É possível saber se sua conta é membro de uma organização no AWS Organizations. Os membros da organização podem ser afetados por uma SCP. Para visualizar esses dados usando o comando da AWS CLI ou operação de API da AWS, você deve ter permissões para a ação `organizations:DescribeOrganization` da sua entidade do AWS Organizations. Você deve ter

permissões adicionais para executar a operação no console do AWS Organizations. Para saber se uma SCP está negando acesso a uma solicitação específica ou alterar as permissões efetivas, entre em contato com o administrador do AWS Organizations.

Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões para uma sessão vêm da entidade (usuário ou função) do IAM usada para criar a sessão e da política da sessão. As permissões de política baseada em identidade da entidade são limitadas pela política de sessão e pelo limite de permissões. As permissões efetivas para esse conjunto de tipos de política são a interseção de todos os três tipos de política. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações as políticas de sessão, consulte Políticas de Sessão.



Delegar responsabilidade para outras pessoas usando limites de permissões

Você pode usar limites de permissões para delegar tarefas de gerenciamento de permissões, como a criação de usuários, aos usuários do IAM em sua conta. Isso permite que outras pessoas executem tarefas em seu nome com um limite específico de permissões.

Por exemplo, suponha que María seja a administradora da da X-Company Conta da AWS. Ela quer delegar as tarefas de criação de usuários para Zhang. No entanto, ela deve garantir que Zhang crie usuários que sigam as seguintes regras da empresa:

- Os usuários não podem usar o IAM para criar ou gerenciar usuários, grupos, funções ou políticas.

- Os usuários têm acesso negado ao bucket `logs` do Amazon S3 e não podem acessar a instância `i-1234567890abcdef0` do Amazon EC2.
- Os usuários não podem remover suas próprias políticas de limite.

Para impor essas regras, María executa as seguintes tarefas, para as quais os detalhes são incluídos a seguir:

1. María cria a política gerenciada `XCompanyBoundaries` para uso como um limite de permissões para todos os novos usuários na conta.
2. María cria a política gerenciada `DelegatedUserBoundary` e atribui-a como o limite de permissões a Zhang. Maria anota o ARN de seu usuário do admin e o usa na política para impedir que Zhang o acesse.
3. María cria a política gerenciada `DelegatedUserPermissions` e anexa-a como uma política de permissões a Zhang.
4. María informa Zhang sobre suas novas responsabilidades e limitações.

Tarefa 1: María deve primeiro criar uma política gerenciada para definir o limite para os novos usuários. María permitirá que Zhang forneça aos usuários as políticas de permissões de que precisam, mas ela deseja que esses usuários sejam restringidos. Para fazer isso, ela cria a seguinte política gerenciada pelo cliente com o nome `XCompanyBoundaries`. Essa política faz o seguinte:

- Permite que os usuários tenham acesso total a vários serviços
- Permite acesso autogerenciado limitado no console do IAM. Isso significa que eles podem alterar a senha depois de fazer login no console. Eles não podem definir a senha inicial. Para permitir isso, adicione a ação `"*LoginProfile"` à instrução `AllowManageOwnPasswordAndAccessKeys`.
- Nega aos usuários o acesso ao bucket de logs do Amazon S3 ou à instância `i-1234567890abcdef0` do Amazon EC2

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
```

```
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
```

```
        }
    ]
}
```

Cada instrução tem uma finalidade diferente:

1. A instrução `ServiceBoundaries` dessa política permite acesso total aos serviços especificados da AWS. Isso significa que as ações de um novo usuário nesses serviços são limitadas apenas pelas políticas de permissões que são anexadas ao usuário.
2. A instrução `AllowIAMConsoleForCredentials` permite acesso para listar todos os usuários do IAM. Esse acesso é necessário para navegar na página Usuários no Console de gerenciamento da AWS. Ele também permite visualizar os requisitos de senha da conta, que é necessário ao alterar sua própria senha.
3. A instrução `AllowManageOwnPasswordAndAccessKeys` permite que os usuários gerenciem apenas suas próprias chaves de acesso programático e a senha do console. Isso é importante se Zhang ou outro administrador atribuir a um novo usuário uma política de permissões com acesso total ao IAM. Nesse caso, esse usuário pode alterar suas próprias permissões ou as de outros usuários. Essa instrução impede que isso ocorra.
4. A instrução `DenyS3Logs` nega explicitamente o acesso ao bucket de `logs`.
5. A instrução `DenyEC2Production` nega explicitamente o acesso à instância de `i-1234567890abcdef0`.

Tarefa 2: María deseja permitir que Zhang crie todos os usuários da X-Company, mas apenas com o limite de permissões `XCompanyBoundaries`. Ela cria a seguinte política gerenciada pelo cliente chamada `DelegatedUserBoundary`. Essa política define o número máximo de permissões que Zhang pode ter.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
```

```
                "iam:ListPolicyVersions",
                "iam:ListRolePolicies",
                "iam:ListSSHPublicKeys",
                "iam:ListServiceSpecificCredentials",
                "iam:ListSigningCertificates",
                "iam:ListUserPolicies",
                "iam:ListUsers",
                "iam:SetDefaultPolicyVersion",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy",
                "iam:UpdateGroup",
                "iam:UpdateLoginProfile",
                "iam:UpdateUser"
            ],
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Cada instrução tem uma finalidade diferente:

1. A instrução `CreateOrChangeOnlyWithBoundary` permite que Zhang crie usuários do IAM, mas somente se ele usar a politica `XCompanyBoundaries` para definir o limite de permissões. Essa instrução também permite que ele defina o limite de permissões para os usuários existentes, mas apenas usando a mesma política. Finalmente, essa instrução permite que Zhang gerencie políticas de permissões para usuários com esse limite de permissões definido.
2. A instrução `CloudWatchAndOtherIAMTasks` permite que Zhang conclua tarefas de gerenciamento de outro usuário, grupo e política. Ele tem permissões para redefinir senhas e criar chaves de acesso para qualquer usuário do IAM não listado no elemento `NotResource` da política. Isso permite a ele ajudar os usuários com problemas de login.
3. A instrução `NoBoundaryPolicyEdit` nega a Zhang o acesso para atualizar a política `XCompanyBoundaries`. Ele não tem permissão para alterar nenhuma política usada para definir o limite de permissões para si mesmo ou para outros usuários.
4. A instrução `NoBoundaryUserDelete` nega a Zhang o acesso para excluir o limite de permissões para si mesmo ou outros usuários.

Em seguida, María atribui a política `DelegatedUserBoundary` como o limite de permissões para o usuário Zhang.

Tarefa 3: como o limite de permissões limita o número máximo de permissões, mas não concede acesso por conta própria, Maria deve criar uma política de permissões para Zhang. Ela cria a seguinte política chamada `DelegatedUserPermissions`. Esta política define as operações que Zhang pode executar, dentro do limite definido.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
```

```
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
    ]
}
```

Cada instrução tem uma finalidade diferente:

1. A instrução `IAM` da política permite a Zhang acesso total ao IAM. No entanto, como seu limite de permissões permite apenas algumas operações do IAM, suas permissões efetivas do IAM são limitadas apenas pelo seu limite de permissões.
2. A instrução `CloudWatchLimited` permite que Zhang execute cinco ações no CloudWatch. Seu limite de permissões permite todas as ações no CloudWatch, portanto, suas permissões efetivas do CloudWatch são limitadas apenas por sua política de permissões.
3. A instrução `S3BucketContents` permite que Zhang liste o bucket `ZhangBucket` do Amazon S3. No entanto, seu limite de permissões não permite nenhuma ação do Amazon S3, portanto, ele não pode executar nenhuma operação do S3, independentemente de sua política de permissões.

> **Note**
>
> As políticas de Zhang permitem que ele crie um usuário que acessa recursos do Amazon S3 que ele não pode acessar. Ao delegar essas ações administrativas, Maria efetivamente confia a Zhang o acesso ao Amazon S3.

Em seguida, María anexa a política `DelegatedUserPermissions` como a política de permissões para o usuário `Zhang`.

Tarefa 4: Ela fornece a Zhang as instruções para criar um novo usuário. Ela informa que ele pode criar novos usuários com qualquer permissão que eles precisem, mas deve atribuir a eles a política `XCompanyBoundaries` como um limite de permissões.

Zhang executa as seguintes tarefas:

1. Zhang cria um usuário com o Console de gerenciamento da AWS. Ele digita o nome do usuário `Nikhil` e permite acesso ao console para o usuário. Ele desmarca a caixa de seleção ao lado de Requires password reset (Requer redefinição de senha), porque as políticas acima permitem que os usuários alterem a senha somente depois que fizerem login no console do IAM.
2. Na página Set permissions (Definir permissões), Zhang escolhe as políticas de permissões IAMFullAccess e AmazonS3ReadOnlyAccess que permitem que Nikhil faça seu trabalho.
3. Zhang ignora a seção Set permissions boundary (Definir limite de permissões) esquecendo as instruções de María.
4. Zhang revisa os detalhes do usuário e seleciona Create user (Criar usuário).

   A operação falha e o acesso é negado. O limite de permissões `DelegatedUserBoundary` de Zhang exige que qualquer usuário que ele crie tenha a política `XCompanyBoundaries` usada como um limite de permissões.
5. Zhang retorna à página anterior. Na seção Set permissions boundary (Definir limite de permissões), ele escolhe a política `XCompanyBoundaries`.
6. Zhang revisa os detalhes do usuário e seleciona Create user (Criar usuário).

   O usuário é criado.

Quando Nikhil faz login, ele tem acesso ao IAM e ao Amazon S3, com exceção das operações que são negadas pelo limite de permissões. Por exemplo, ele pode alterar sua própria senha no IAM, mas não pode criar outro usuário ou editar suas políticas. Nikhil tem acesso somente leitura ao Amazon S3.

Se alguém adiciona uma política baseada em recurso ao bucket do `logs` que permite que Nikhil coloque um objeto no bucket, ele ainda não pode acessar o bucket. O motivo é que as ações no bucket `logs` são explicitamente negadas por seu limite de permissões. Uma negação explícita em qualquer tipo de política resulta em uma solicitação negada.

No entanto, se uma política baseada em recurso anexada a um segredo do Secrets Manager permitir que Nikhil execute a ação `secretsmanager:GetSecretValue`, Nikhil poderá recuperar e

descriptografar o segredo. O motivo é que as operações do Secrets Manager não são explicitamente negadas por seu limite de permissões, e as negações implícitas nos limites de permissões não limitam as políticas baseadas em recurso.

Políticas baseadas em identidade e em recurso

Uma política é um objeto na AWS que, quando associado a uma identidade ou recurso, define suas permissões. Quando você cria uma política de permissões para restringir o acesso a um recurso, você pode escolher uma política baseada em identidade ou uma política baseada em recurso.

As políticas baseadas em identidade são anexadas a um usuário, grupo ou função do IAM. Essas políticas permitem que você especifique o que cada identidade pode fazer (suas respectivas permissões). Por exemplo, você pode anexar a política ao usuário do IAM chamado John, informando que ele tem permissão para executar a ação `RunInstances` do Amazon EC2. A política poderia afirmar ainda que John tem permissão para obter itens de uma tabela do Amazon DynamoDB chamada `MyCompany`. Você também pode permitir que John gerencie suas próprias credenciais de segurança do IAM. As políticas baseadas em identidade podem ser em linha ou gerenciadas.

Políticas baseadas em recurso são anexadas a um recurso. Por exemplo, é possível anexar políticas baseadas em recurso a buckets do Amazon S3, filas do Amazon SQS, endpoints da VPC, chaves de criptografia do AWS Key Management Service e tabelas e fluxos do Amazon DynamoDB e recursos de login da AWS. Para obter uma lista de serviços que oferecem suporte a políticas baseadas em recursos, consulte AWSServiços da que funcionam com o IAM.

Com as políticas baseadas em recursos, você pode especificar quem tem acesso ao recurso e quais ações essas pessoas podem realizar nele. Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?. As políticas baseadas em recursos são apenas em linha, não gerenciadas.

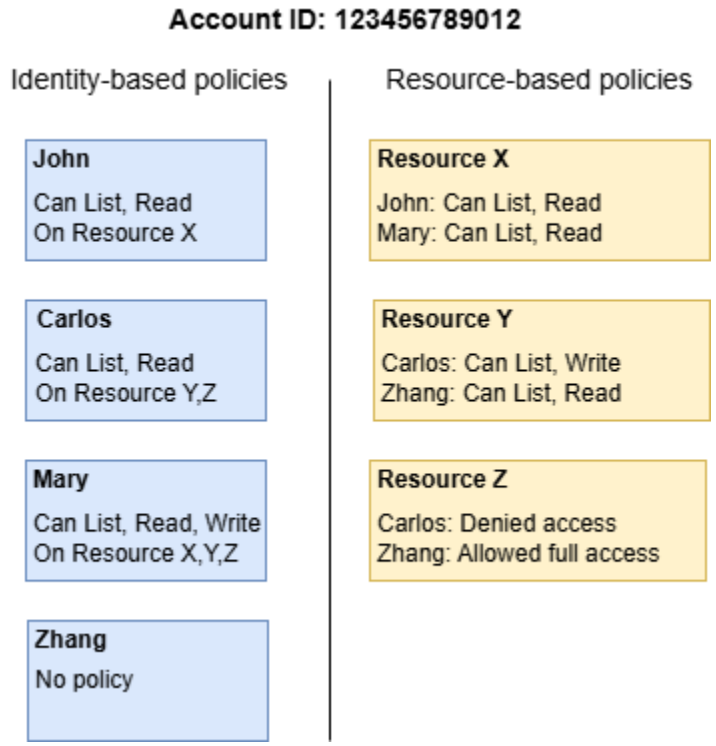
> **Note**
>
> Políticas baseadas em recursos são diferentes das permissões no nível do recurso. Você pode anexar as políticas baseadas em recursos diretamente em um recurso, como indicado neste tópico. As permissões no nível do recurso referem-se à capacidade de usar ARNs para especificar recursos individuais em uma política. As políticas baseadas em recursos são compatíveis apenas com alguns serviços da AWS. Para obter uma lista de serviços que são

compatíveis com políticas baseadas em recursos e permissões no nível do recurso, consulte AWSServiços da que funcionam com o IAM.

Para saber como as políticas baseadas em identidade e as baseadas em recurso interagem na mesma conta, consulte Avaliação de políticas para solicitações em uma única conta.

Para saber como as políticas interagem entre contas, consulte Lógica de avaliação de política entre contas.

Para compreender melhor esses conceitos, exiba a figura a seguir. O administrador da conta `123456789012` anexou políticas baseadas na identidade aos usuários `John`, `Carlos` e `Mary`. Algumas das ações nessas políticas podem ser realizadas em recursos específicos. Por exemplo, o usuário `John` pode realizar algumas ações em `Resource X`. Esta é uma permissão no nível do recurso em uma política baseada em identidade. O administrador também adicionou políticas baseadas em recurso a `Resource X`, `Resource Y` e `Resource Z`. As políticas baseadas em recurso permitem especificar quem pode acessar esse recurso. Por exemplo, a política baseada em recurso em `Resource X` permite a lista de usuários `John` e `Mary` e o acesso de leitura ao recurso.



O exemplo de conta `123456789012` permite que os seguintes usuários realizem as ações listadas:

- John: John pode realizar as ações de listar e ler no `Resource X`. Ele recebe essa permissão pela política baseada em identidade pelo usuário e a política baseada em recurso em `Resource X`.
- Carlos: Carlos pode realizar as ações de listar, ler e gravar no `Resource Y`, mas seu acesso ao `Resource Z` é negado. A política baseada em identidade em Carlos permite que ele realize ações de listagem e leitura em `Resource Y`. A política baseada em recurso `Resource Y` também dá a ele permissões de gravação. No entanto, embora a política baseada em identidade dê a ele acesso a `Resource Z`, a política baseada em recurso `Resource Z` nega esse acesso. Uma `Deny` explícita substitui uma `Allow`, e o acesso a `Resource Z` é negado. Para obter mais informações, consulte Lógica da avaliação de política.
- Mary: Mary pode realizar as operações de listar, ler e gravar no `Resource X`, no `Resource Y` e no `Resource Z`. A política baseada em identidade permite mais ações em mais recursos do que as políticas baseadas em recursos, mas nenhuma delas nega acesso.
- Zhang: Zhang tem total acesso ao `Resource Z`. Zhang não tem políticas baseadas em identidade, mas a política baseada em recurso `Resource Z` dá a ele acesso total ao recurso. Zhang também pode executar ações de listagem e leitura no `Resource Y`.

As políticas baseadas em identidade e as baseadas em recurso são políticas de permissões e avaliadas juntas. Para uma solicitação a que somente políticas de permissões se apliquem, a AWS primeiro verifica todas as políticas de um `Deny`. Caso haja alguma, a solicitação é negada. Em seguida, o AWS verifica cada `Allow`. Se, pelo menos uma declaração de política permitir a ação na solicitação, a solicitação será permitida. Não importa se a `Allow` está na política baseada em identidade ou na política baseada em recurso.

> **Important**
>
> Essa lógica só se aplica quando a solicitação é feita em uma única Conta da AWS. Para solicitações feitas de uma conta para outra, o solicitante em `Account A` deve ter uma política baseada em identidade que permita fazer uma solicitação para o recurso em `Account B`. Além disso, a política baseada em recurso no `Account B` deve permitir o solicitante em `Account A` para acessar o recurso. Deve haver políticas em ambas as contas que permitam a operação, caso contrário, a solicitação falhará. Para obter mais informações sobre políticas baseadas em recursos para acesso entre contas, consulte Acesso a recursos entre contas no IAM.

Um usuário com permissões específicas pode solicitar um recurso que também tenha uma política de permissões anexada a ele. Nesse caso, a AWS avalia os dois conjuntos de permissões ao determinar se concede ou não acesso ao recurso. Para obter informações sobre como as políticas são avaliadas, consulte Lógica da avaliação de política.

> **Note**
>
> O Amazon S3 oferece suporte às políticas baseadas em identidade e às políticas baseadas em recurso (referenciadas como políticas de bucket). Além disso, o Amazon S3 oferece suporte a um mecanismo de permissões conhecido como uma lista de controle de acesso (ACL) que não depende de políticas e permissões do IAM. Você pode usar políticas do IAM em combinação com as ACLs do Amazon S3. Para mais informações, acesse Controle de acesso no Guia do usuário do Amazon Simple Storage Service.

Controlar o acesso a recursos da AWS usando as políticas

Você pode usar uma política para controlar o acesso a recursos no IAM ou em toda a AWS.

Para usar uma política para controlar o acesso na AWS, você deve entender como a AWS concede acesso. A AWS é composta de coleções de recursos. Um usuário do IAM é um recurso. Um bucket do Amazon S3 é um recurso. Quando usa a API da AWS, a AWS CLI ou o Console de gerenciamento da AWS para executar uma ação (como a criação de um usuário), você envia uma solicitação para essa operação. Sua solicitação especifica uma ação, um recurso, uma entidade principal (usuário ou função), uma conta principal e qualquer informação necessária. Todas essas informações fornecem o contexto.

AWSO verifica se você (a entidade principal) está autenticado (cadastrado) e autorizado (tem permissão) para executar a ação especificada no recurso especificado. Durante a autorização, o AWS verifica todas as políticas aplicáveis ao contexto da sua solicitação. A maioria das políticas é armazenada no AWS, como documentos JSON e especifica as permissões para as entidades principais. Para obter mais informações sobre os tipos e os usos de políticas, consulte Políticas e permissões no AWS Identity and Access Management.

AWSA autorizará a solicitação somente se cada parte de sua solicitação tiver permissão concedida pelas políticas. Para visualizar um diagrama desse processo, consulte Como o IAM funciona. Para obter detalhes sobre como a AWS determina se uma solicitação deve obter permissão, consulte Lógica da avaliação de política.

Ao criar uma política do IAM, você pode controlar o acesso ao seguinte:

- Entidades de segurança: controle o que a pessoa que faz a solicitação (a entidade de segurança) tem permissão para fazer.
- Identidades do IAM: controle quais identidades (usuários, perfis e grupos do IAM) podem ser acessadas e como.
- Políticas do IAM: controle quem pode criar, editar e excluir políticas gerenciadas pelo cliente e quem pode anexar e desvincular todas as políticas gerenciadas.
- Recursos da AWS: controle quem tem acesso aos recursos usando uma política baseada em identidade ou uma política baseada em recurso.
- Contas da AWS: controle se uma solicitação tem permissão apenas para membros de uma conta específica.

Com as políticas, você pode especificar quem tem acesso aos recursos da AWS e quais ações essas pessoas podem realizar neles. Cada usuário do IAM começa sem permissões. Em outras palavras, por padrão, os usuários não podem fazer nada, nem mesmo visualizar suas próprias chaves de acesso. Para dar permissão a um usuário para fazer algo, você pode adicionar a permissão para esse usuário, ou seja, anexar uma política ao usuário. Ou você pode adicionar o usuário a um grupo de usuários que tenha a permissão pretendida.

Por exemplo, você pode conceder a um usuário permissão para listar suas próprias chaves de acesso. Você também pode expandir essa permissão e também permitir que cada usuário crie, atualize e exclua suas próprias chaves.

Quando você concede permissões a um grupo de usuários, todos os usuários nesse grupo de usuários obtêm essas permissões. Por exemplo, você pode dar ao grupo de usuários Administradores permissão para executar qualquer uma das ações do IAM em qualquer um dos recursos da Conta da AWS. Outro exemplo: você pode dar ao grupo de usuários Gerentes permissão para descrever as instâncias do Amazon EC2 da Conta da AWS.

Para obter informações sobre como delegar permissões básicas aos usuários, perfis e grupos do IAM, consulte Permissões necessárias para acessar recursos do IAM. Para obter exemplos adicionais de políticas que ilustram permissões básicas, consulte Exemplos de política para administrar recursos do IAM.

Controlar o acesso de entidades principais do

Você pode usar as políticas para controlar o que a pessoa que está fazendo a solicitação (a entidade principal) pode fazer. Para fazer isso, você deve anexar uma política baseada em identidade à identidade da pessoa (usuário, grupo de usuários ou função). Você também pode usar um limite de permissões para definir as permissões máximas que uma entidade (usuário ou função) pode ter.

Por exemplo, suponha que você queira que o usuário Zhang Wei tenha acesso total ao CloudWatch, Amazon DynamoDB, Amazon EC2 e Amazon S3. Você pode criar duas políticas diferentes para que possa separá-las posteriormente se precisar de um conjunto de permissões para outro usuário. Ou você pode colocar as duas permissões em uma única política e, em seguida, anexar essa política ao usuário do IAM que se chama Zhang Wei. Você também pode anexar uma política a um grupo de usuários ao qual Zhang pertence, ou a uma função que Zhang pode assumir. Como resultado, quando Zhang visualiza o conteúdo de um bucket do S3, suas solicitações são permitidas. Se ele tentar criar um novo usuário do IAM, a solicitação será negada pois ele não tem permissão.

Você pode usar um limite de permissões para Zhang para ter certeza de que ele nunca terá acesso ao bucket `amzn-s3-demo-bucket1` do S3. Para fazer isso, determine o número máximo de permissões que deseja que Zhang tenha. Nesse caso, você controla o que ele faz usando suas políticas de permissões. Aqui, você só se importa com que ele não tenha acesso ao bucket confidencial. Portanto, você usa a política a seguir para definir o limite de Zhang para permitir todas as ações da AWS para o Amazon S3 e alguns outros serviços, mas negar acesso ao bucket `amzn-s3-demo-bucket1` do S3. Como o limite de permissões não permite nenhuma ação do IAM, ele impede que Zhang exclua seu limite (ou de alguém).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket1",
                "arn:aws:s3:::amzn-s3-demo-bucket1/*"
            ]
        }
    ]
}
```

Quando você atribui uma política como essa como um limite de permissões a um usuário, lembre-se de que ela não concede permissões. Ela define o número máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM. Para obter mais informações sobre esses limites de permissões, consulte Limites de permissões para entidades do IAM.

Para obter informações detalhadas sobre os procedimentos mencionados anteriormente, consulte estes recursos:

- Para saber mais sobre como criar uma política do IAM que possa ser anexada a uma entidade de segurança, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.
- Para saber mais sobre como anexar uma política do IAM a uma entidade de segurança, consulte Adicionar e remover permissões de identidade do IAM.
- Para ver um exemplo de política que concede acesso total ao EC2, consulte Amazon EC2: permite acesso total ao EC2 dentro de uma região específica, de forma programática e no console.
- Para permitir o acesso somente leitura a um bucket do S3, use as duas primeiras instruções do seguinte exemplo de política: Amazon S3: permite acesso de leitura e gravação a objetos em um bucket do S3 de forma programática e no console.
- Para ver um exemplo de política que permite que os usuários definam suas credenciais, como a senha do console, chaves de acesso programático e dispositivos MFA, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.

Controle de acesso a identidades

Você pode usar políticas do IAM para controlar o que os usuários podem fazer em uma identidade criando uma política que você anexa a todos os usuários por meio de um grupo de usuários. Para fazer isso, crie uma política que limita o que pode ser feito em uma identidade ou quem pode acessá-la.

Por exemplo, você pode criar um grupo de usuários chamado AllUsers e, em seguida, anexá-lo a todos os usuários. Ao criar o grupo de usuários, é possível conceder acesso a todos os usuários para que eles definam suas credenciais conforme descrito na seção anterior. Em seguida, você pode criar uma política que negue o acesso para alterar o grupo de usuários, a menos que o nome do usuário seja incluído na condição da política. Mas essa parte da política nega acesso a qualquer pessoa, com exceção dos usuários listados. Também é necessário incluir permissões para permitir todas as ações de gerenciamento do grupo de usuários para todos no grupo de usuários. Por fim, anexe essa política ao grupo de usuários para que ela se aplique a todos os usuários. Dessa forma, quando um usuário não especificado na política tentar fazer alterações no grupo de usuários, a solicitação será negada.

Para criar esta política com o editor visual

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
3. Escolha Criar política.
4. Na seção Editor de políticas, escolha a opção Visual.
5. Em Selecionar um serviço, escolha IAM.
6. Em Ações permitidas, digite **`group`** na caixa de pesquisa. O editor visual mostra todas as ações do IAM que contêm a palavra `group`. Marque todas as caixas de seleção.
7. Escolha Recursos para especificar os recursos para a sua política. Com base nas ações que escolheu, você deverá ver os tipos de recursos grupo e usuário.

   - grupo: escolha Adicionar ARN. Em Recurso em, selecione a opção Qualquer conta. Marque a caixa de seleção Qualquer nome de grupo com caminho e digite o nome do grupo de usuários **`AllUsers`**. Selecione Adicionar ARNs.

- usuário: marque a caixa de seleção ao lado de Qualquer um nessa conta.

Uma das ações que você escolheu, `ListGroups`, não oferece suporte ao uso de recursos específicos. Você não precisa escolher Todos os recursos para essa ação. Quando você salva a política ou visualiza a política no editor de JSON, pode ver que o IAM cria automaticamente um novo bloco de permissões concedendo permissão para essa ação em todos os recursos.

8. Para adicionar outro bloco de permissões, escolha Adicionar mais permissões.
9. Escolha Selecionar um serviço e depois escolha IAM.
10. Escolha Ações permitidas e depois escolha Mudar para negar permissões. Ao fazer isso, o bloco inteiro é usado para negar permissões.
11. Digite **`group`** na caixa de pesquisa. O editor visual mostra todas as ações do IAM que contêm a palavra `group`. Marque as caixas de seleção ao lado das seguintes ações:

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy
    - DetachGroupPolicy
    - PutGroupPolicy
    - UpdateGroup

12. Escolha Recursos para especificar os recursos para a sua política. Com base nas ações que escolheu, você deve ver o tipo de recurso grupo. Escolha Add ARNs. Em Recurso em, selecione a opção Qualquer conta. Em Qualquer nome de grupo com caminho digite o nome do grupo de usuários **`AllUsers`**. Selecione Adicionar ARNs.
13. Escolha Condições da solicitação - opcional e depois escolha Adicionar outra condição. Preencha o formulário com os valores a seguir:

    - Chave de condição: escolha aws:username
    - Qualifier (Qualificador): escolha Default (Padrão)
    - Operator (Operador): escolha StringNotEquals
    - Valor: digite **`srodriguez`** e escolha Adicionar para adicionar outro valor. Digite **`mjackson`** e escolha Adicionar para adicionar outro valor. Digite **`adesai`** e escolha Adicionar condição.

Essa condição garante que o acesso será negado às ações de gerenciamento do grupo de usuários especificado quando o usuário que está fazendo a chamada não estiver incluído na lista. Como isso nega a permissão de forma explícita, ela substitui o bloco anterior que permitia que esses usuários chamassem as ações. Os usuários da lista não têm acesso negado e recebem permissão no primeiro bloco de permissões para que possam gerenciar totalmente o grupo de usuários.

14. Quando terminar, escolha Próximo.

> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar na opção de editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

15. Na página Revisar e criar, em Nome da política, digite **`LimitAllUserGroupManagement`**. Em Descrição, digite **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**. Revise Permissões definidas nessa política para ter certeza de que você concedeu as permissões que pretendia. Em seguida, escolha Criar política para salvar sua nova política.
16. Anexe a política ao seu grupo de usuários. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

Como alternativa, você pode criar a mesma política usando este exemplo de documento de política JSON. Para visualizar esta política JSON, consulte IAM: permite que os usuários do IAM gerenciem um grupo de forma programática e no console. Para obter instruções detalhadas para criar uma política usando um documento JSON, consulte the section called “Criar políticas usando o editor de JSON”.

Controle de acesso a políticas

Você pode controlar como seus usuários aplicam as políticas gerenciadas da AWS. Para fazer isso, anexe esta política a todos os seus usuários. Idealmente, você pode fazer isso usando um grupo de usuários.

Por exemplo, você pode criar uma política que permita aos usuários anexar apenas as políticas gerenciadas da AWS IAMUserChangePassword e PowerUserAccess a um novo usuário, grupo de usuários ou função do IAM.

Para as políticas gerenciadas pelo cliente, você pode controlar quem pode criar, atualizar e excluir essas políticas. Você pode controlar quem pode anexar políticas às entidades principais (usuários, perfis e grupos do IAM) e quem pode desanexá-las. Você também pode controlar quais políticas um usuário pode anexar e desanexar de quais entidades.

Por exemplo, você pode atribuir permissões a um administrador de conta para criar, atualizar e excluir políticas. Em seguida, você atribui permissões a um líder de equipe ou outro administrador limitado para anexar e desanexar essas políticas às entidades principais que o administrador limitado gerencia.

Para obter mais informações, consulte estes recursos:

- Para saber mais sobre como criar uma política do IAM que possa ser anexada a uma entidade de segurança, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.
- Para saber mais sobre como anexar uma política do IAM a uma entidade de segurança, consulte Adicionar e remover permissões de identidade do IAM.
- Para ver um exemplo de política para limitar o uso de políticas gerenciadas, consulte IAM: limita as políticas gerenciadas que podem ser aplicadas a um usuário, grupo ou função do IAM.

Controle de permissões para a criação, atualização e exclusão de políticas gerenciadas pelo cliente

Você pode usar políticas do IAM para controlar quem tem permissão para criar, atualizar e excluir as políticas gerenciadas pelo cliente na sua Conta da AWS. A lista a seguir contém operações de API que fazem parte diretamente da criação, atualização e exclusão de políticas ou versões de política:

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

As operações da API na lista anterior correspondem a ações que você pode permitir ou negar ou seja, permissões que você pode conceder usando uma política do IAM.

Considere a seguinte política de exemplo. Ela permite que um usuário crie, atualize (isto é, crie uma nova versão da política), exclua e defina uma versão padrão para todas as políticas gerenciadas pelo cliente na Conta da AWS. O exemplo de política também permite que o usuário liste as políticas e obtenha políticas. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

Example Exemplo de política que permite criar, atualizar, excluir, listar, obter e definir a versão padrão para todas as políticas

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

Você pode criar políticas que limitam o uso dessas operações da API para afetar somente as políticas gerenciadas que você especificar. Por exemplo, talvez você queira permitir que um usuário defina a versão padrão e exclua versões de políticas, mas somente para políticas específicas gerenciadas pelo cliente. Isso é feito especificando-se o ARN da política no elemento `Resource` da política que concede essas permissões.

O exemplo a seguir mostra uma política que permite que um usuário exclua versões de política e defina a versão padrão. Mas essas ações são permitidas somente para as políticas gerenciadas pelo cliente que incluem o caminho /TEAM-A/. O ARN da política gerenciada pelo cliente é especificado no elemento `Resource` da política. (Neste exemplo, o ARN inclui um caminho e um caractere curinga e, portanto, corresponde a todas as políticas gerenciadas pelo cliente que incluem o caminho /TEAM-A/). Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

Para obter mais informações sobre o uso de caminhos nos nomes de políticas gerenciadas pelo cliente, consulte Nomes amigáveis e caminhos.

Example Exemplo de política que permite excluir versões de política e definir a versão padrão somente para políticas específicas

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:DeletePolicyVersion",
            "iam:SetDefaultPolicyVersion"
        ],
        "Resource": "arn:aws:iam::111122223333:policy/TEAM-A/*"
    }
}
```

Controle de permissões para anexar e desvincular políticas gerenciadas

Você também pode usar as políticas do IAM para permitir que os usuários trabalhem apenas com políticas gerenciadas específicas. Em suma, você pode controlar quais permissões um usuário pode conceder a outras entidades principais.

A lista a seguir mostra as operações da API que fazem parte diretamente da anexação e desanexação de políticas gerenciadas às entidades principais:

- AttachGroupPolicy
- AttachRolePolicy

- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Você pode criar políticas que limitam o uso dessas operações da API para afetar somente as políticas gerenciadas e/ou as entidades principais que você especifica. Por exemplo, talvez você queira permitir que um usuário anexe políticas gerenciadas, mas somente as políticas gerenciadas que você especificar. Ou, talvez você queira permitir que um usuário anexe políticas gerenciadas, mas somente às entidades principais que você especificar.

O exemplo de política a seguir permite que um usuário só anexe políticas gerenciadas a perfis e grupos do IAM que incluam o caminho /TEAM-A/. Os ARNs do grupo de usuários e da função são especificados no elemento `Resource` da política. (Neste exemplo, os ARNs incluem um caminho e um caractere curinga e, por isso, correspondem a todos os perfis e grupos do IAM que incluem o caminho /TEAM-A/). Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

Example Exemplo de política que permite anexar políticas gerenciadas somente a grupos de usuários ou funções específicos

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ]
    }
}
```

Você pode limitar ainda mais as ações no exemplo anterior para afetar somente políticas específicas. Ou seja, você pode controlar quais permissões um usuário pode anexar a outras entidades de segurança, adicionando uma condição à política.

No exemplo a seguir, a condição garante que as permissões `AttachGroupPolicy` e `AttachRolePolicy` sejam concedidas somente quando a política sendo anexada corresponda a uma das políticas especificadas. A condição usa a `iam:PolicyARN` chave de condição para determinar qual política (ou políticas) pode ser anexada. A política de exemplo a seguir expande o exemplo anterior. Ela permite que um usuário só anexe as políticas gerenciadas que incluem o caminho /TEAM-A/ aos perfis e grupos do IAM que incluam o caminho /TEAM-A/. Para saber mais sobre como criar uma política usando este exemplo de documento de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:group/TEAM-A/*",
            "arn:aws:iam::111122223333:role/TEAM-A/*"
        ],
        "Condition": {
            "ArnLike": {
                "iam:PolicyARN": "arn:aws:iam::111122223333:policy/TEAM-A/*"
            }
        }
    }
}
```

Esta política usa o operador de condição `ArnLike`, mas você também pode usar o operador de condição `ArnEquals` porque esses dois operadores de condição se comportam de forma idêntica. Para obter mais informações sobre `ArnLike` e `ArnEquals`, consulte Operadores de condição de

nome do recurso da Amazon (ARN) na seção Tipos de condição da Referência sobre elementos de políticas.

Por exemplo, é possível limitar o uso dessas ações para envolver somente as políticas gerenciadas que você especifica. Isso é feito especificando-se o ARN da política no elemento `Condition` da política que concede essas permissões. Por exemplo, para especificar o ARN de uma política gerenciada pelo cliente:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

Você pode especificar o ARN de uma política gerenciada da AWS no elemento `Condition` da política. O ARN de uma política gerenciada da AWS usa o alias especial `aws` no ARN da política, em vez de um ID de conta, como neste exemplo:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Controlar o acesso aos recursos

Você pode controlar o acesso aos recursos usando uma política baseada em identidade ou em recurso. Em uma política baseada em identidade, você anexa a política a uma identidade e especifica que recursos essa identidade pode acessar. Em uma política baseada em recursos, você anexa uma política ao recurso que deseja controlar. Na política, você especifica quais entidades principais podem acessar esse recurso. Para obter mais informações sobre ambos os tipos de políticas, consulte Políticas baseadas em identidade e em recurso.

Para obter mais informações, consulte estes recursos:

- Para saber mais sobre como criar uma política do IAM que possa ser anexada a uma entidade de segurança, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.
- Para saber mais sobre como anexar uma política do IAM a uma entidade de segurança, consulte Adicionar e remover permissões de identidade do IAM.
- O Amazon S3 oferece suporte ao uso de políticas baseadas em recurso em seus buckets. Para obter mais informações, consulte Exemplos de políticas de buckets.

Criadores de recursos não têm permissões automaticamente

Se você fizer login usando as credenciais do Usuário raiz da conta da AWS, terá permissão para executar qualquer ação nos recursos que pertencem à conta. No entanto, isso não é verdadeiro para usuários do IAM. Um usuário do IAM pode receber acesso para criar um recurso, mas as permissões do usuário, mesmo para esse recurso, são limitadas ao que foi explicitamente concedido. Isso significa que só pelo fato de criar um recurso, como uma função do IAM, você não tem automaticamente permissão para editar ou excluir essa função. Além disso, sua permissão pode ser revogada a qualquer momento pelo proprietário da conta ou por outro usuário que tenha recebido acesso para gerenciar suas permissões.

Controle de acesso a entidades de segurança em uma conta específica

Você pode conceder diretamente aos usuários do IAM em sua própria conta acesso aos seus recursos. Se os usuários de outra conta precisarem de acesso a seus recursos, você poderá criar uma função do IAM. Uma função é uma entidade que inclui permissões, mas que não está associada a um usuário específico. Os usuários de outras contas podem, então, assumir a função e acessar os recursos de acordo com as permissões que você tiver atribuído à função. Para obter mais informações, consulte Acesso a um usuário do IAM em outra Conta da AWS de sua propriedade.

> **Note**
>
> Alguns serviços oferecem suporte a políticas baseadas em recurso, conforme descrito em Políticas baseadas em identidade e em recurso (como o Amazon S3, Amazon SNS e Amazon SQS). Para esses serviços, uma alternativa ao uso de funções é anexar uma política ao recurso (bucket, tópico ou fila) que você deseja compartilhar. A política baseada em recurso pode especificar a conta da AWS com permissões para acessar o recurso.

Controle de acesso para usuários e funções do IAM usando etiquetas

Use as informações da seção a seguir para controlar quem pode acessar os usuários e as funções do IAM e quais recursos os usuários e as funções podem acessar. Para obter mais informações gerais e exemplos de controle de acesso a outros recursos da AWS, incluindo recursos do IAM, consulte Tags para recursos do AWS Identity and Access Management.

> **Note**
>
> Para obter detalhes sobre a distinção entre maiúsculas e minúsculas para chaves de tag e valores de chave de tag, consulte Case sensitivity.

As etiquetas podem ser anexadas ao recurso do IAM, transmitidas na solicitação, ou anexadas à entidade de segurança que está fazendo a solicitação. Um usuário ou uma função do IAM pode ser um recurso e uma entidade de segurança. Por exemplo, é possível escrever uma política que permita que um usuário liste os grupos de um usuário. Essa operação é permitida somente se o usuário que está fazendo a solicitação (o principal) tiver a mesma tag `project=blue` que o usuário que está tentando visualizar. Neste exemplo, o usuário pode visualizar a associação ao grupo de qualquer usuário, incluindo ele mesmo, desde que esteja trabalhando no mesmo projeto.

Para controlar o acesso com base em tags, forneça informações sobre a tag no elemento de condição de uma política. Ao criar uma política do IAM, você pode usar etiquetas do IAM e a chave de condição da etiqueta associada para controlar o acesso a qualquer um das seguintes opções:

- Recurso: controle o acesso aos recursos do usuário ou da função com base em suas etiquetas. Para fazer isso, use a chave de condição aws:ResourceTag/***key-name*** para especificar o par de chave-valor da tag a ser associado ao recurso. Para obter mais informações, consulte Controlar o acesso aos recursos do AWS.
- Solicitação: controle quais etiquetas podem ser transmitidas em uma solicitação do IAM. Para isso, use a chave de condição aws:RequestTag/***key-name*** para especificar quais etiquetas podem ser adicionadas, alteradas ou removidas de um usuário ou de uma função do IAM. Essa chave é usada da mesma forma para recursos do IAM e outros recursos da AWS. Para obter mais informações, consulte Controlar o acesso durante solicitações do AWS.
- Entidade de segurança: controle o que a pessoa que está fazendo a solicitação (a entidade de segurança) tem permissão para fazer com base nas etiquetas anexadas ao usuário ou à função do IAM dessa pessoa. Para isso, use a chave de condição aws:PrincipalTag/***key-name*** para especificar quais etiquetas devem ser anexadas ao usuário ou à função do IAM até a solicitação ser permitida.
- Qualquer parte do processo de autorização: use a chave de condição aws:TagKeys para controlar se chaves de tag específicas podem ser usadas em uma solicitação ou por uma entidade principal. Neste caso, o valor da chave não importa. Essa chave se comporta de forma semelhante para o IAM e outros serviços da AWS. No entanto, quando você etiqueta um usuário no IAM, isso também

controla se a entidade de segurança pode fazer a solicitação para qualquer serviço. Para obter mais informações, consulte Controlar o acesso com base em chaves de tag.

É possível criar uma política do IAM com o editor visual, usando JSON ou importando uma política gerenciada existente. Para obter detalhes, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.

> **Note**
>
> Também é possível passar tags de sessão ao assumir um perfil do IAM ou federar um usuário. Elas são válidas somente durante a sessão.

Controle de acesso de entidades de segurança do IAM

Você pode controlar as ações que o principal tem permissão para realizar com base nas tags associadas à identidade dessa pessoa.

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que qualquer usuário nesta conta visualize a associação ao grupo de qualquer usuário, inclusive a sua própria, desde que esteja trabalhando no mesmo projeto. Essa operação só é permitida quando a tag de recurso do usuário e a tag da entidade principal têm o mesmo valor para a chave de tag `project`. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
```

```
            }
        }]
}
```

Controlar o acesso com base em chaves de tag

Você pode usar tags em suas políticas do IAM para controlar quais chaves de tag específicas podem ser usadas em um recurso ou por uma entidade principal.

Este exemplo mostra como é possível criar uma política baseada em identidade que só permita remover a tag com a chave `temporary` dos usuários. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys":
 ["temporary"]}}
    }]
}
```

Controlar o acesso a recursos da AWS usando tags

Você pode usar etiquetas para controlar o acesso aos recursos da AWS que forem compatíveis com o etiquetamento, incluindo recursos do IAM. Também é possível etiquetar usuários e funções do IAM para controlar o que eles podem acessar. Para saber como etiquetar usuários e funções do IAM, consulte Tags para recursos do AWS Identity and Access Management. Além disso, você pode controlar o acesso aos seguintes recursos do IAM: políticas gerenciadas pelo cliente, provedores de identidade do IAM, perfis de instância, certificados de servidor e dispositivos com MFA virtuais. Para visualizar um tutorial de como criar e testar uma política que permite que funções do IAM com etiquetas de entidade de segurança acessem recursos com etiquetas correspondentes, consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas. Use as

informações na seção a seguir para controlar o acesso a outros recursos da AWS, incluindo recursos do IAM, sem etiquetar usuários ou funções do IAM.

Antes de usar tags para controlar o acesso aos recursos da AWS, você deve entender como a AWS concede acesso. A AWS é composta por conjuntos de recursos. Uma instância do Amazon EC2 é um recurso. Um bucket do Amazon S3 é um recurso. É possível usar a API da AWS, a AWS CLI ou o Console de gerenciamento da AWS para executar uma operação, como a criação de um bucket no Amazon S3. Ao fazer isso, você envia uma solicitação para essa operação. Sua solicitação especifica uma ação, um recurso, uma entidade principal (usuário ou função), uma conta principal e qualquer informação necessária. Todas essas informações fornecem o contexto.

AWSO verifica se você (a entidade principal) está autenticado (cadastrado) e autorizado (tem permissão) para executar a ação especificada no recurso especificado. Durante a autorização, o AWS verifica todas as políticas aplicáveis ao contexto da sua solicitação. A maioria das políticas é armazenada no AWS, como documentos JSON e especifica as permissões para as entidades principais. Para obter mais informações sobre os tipos e os usos de políticas, consulte Políticas e permissões no AWS Identity and Access Management.

AWSA autorizará a solicitação somente se cada parte de sua solicitação tiver permissão concedida pelas políticas. Para visualizar um diagrama e saber mais sobre a infraestrutura do IAM, consulte Como o IAM funciona. Para obter detalhes sobre como o IAM determina se uma solicitação é permitida, consulte Lógica da avaliação de política.

As etiquetas devem ser levadas em consideração nesse processo porque podem ser anexadas ao recurso ou transmitidas na solicitação para serviços que oferecem suporte a marcação. Para controlar o acesso com base em tags, forneça informações sobre a tag no elemento de condição de uma política. Para saber se um serviço da AWS oferece suporte ao controle de acesso usando tags, consulte AWSServiços da que funcionam com o IAM e procure os serviços que contêm Sim na coluna ABAC. Selecione o nome do serviço para visualizar a documentação de controle de acesso e a autorização desse serviço.

Depois, é possível criar uma política do IAM que permite ou nega o acesso a um recurso com base na etiqueta desse recurso. Nessa política, é possível usar as chaves de condição da tag para controlar o acesso a qualquer um dos seguintes:

- Recurso: controle o acesso aos recursos de produtos da AWS com base nas etiquetas desses recursos. Para isso, use a chave de condição aws:ResourceTag/***key-name*** para determinar se o acesso ao recurso será permitido com base nas tags anexadas a ele.

- Solicitação: controle quais etiquetas podem ser transmitidas em uma solicitação. Para fazer isso, use a chave de condição aws:RequestTag/***key-name*** para especificar quais pares de chave-valor de tag podem ser transmitidos em uma solicitação para aplicar uma tag em um recurso da AWS.
- Qualquer parte do processo de autorização: use a chave de condição aws:TagKeys para controlar se as chaves de etiquetas específicas podem estar em uma solicitação.

Você pode criar uma política do IAM visualmente, usando JSON ou importando uma política gerenciada existente. Para obter detalhes, consulte Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente.

> **Note**
>
> Alguns serviços permitem que os usuários especifiquem etiquetas ao criar o recurso, caso tenham permissões para usar a ação que cria o recurso.

Controlar o acesso aos recursos do AWS

É possível usar condições em suas políticas do IAM para controlar o acesso aos recursos da AWS com base nas etiquetas desse recurso. Você pode fazer isso usando a chave de condição global `aws:ResourceTag/`*`tag-key`* ou uma chave específica do serviço. Alguns serviços oferecem suporte apenas à versão dessa chave que é específica do serviço e não à versão global.

> **Warning**
>
> Não tente controlar quem pode passar por uma função marcando a função e, em seguida, usando a chave de condição `ResourceTag` em uma política com a ação `iam:PassRole`. Os resultados dessa abordagem não são confiáveis. Para obter mais informações sobre as permissões necessárias para transmitir uma função a um serviço, consulte Conceda permissões a um usuário para passar um perfil para um serviço da AWS.

Este exemplo mostra como é possível criar uma política baseada em identidade que permita iniciar ou interromper instâncias do Amazon EC2. Essas operações são permitidas apenas se a etiqueta `0wner` da instância tiver o valor do nome do usuário. Esta política define permissões para acesso programático e do console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

É possível anexar essa política aos usuários do IAM na sua conta. Se um usuário chamado `richard` tentar iniciar uma instância do Amazon EC2, a instância deve ser etiquetada como `Owner=richard` ou `owner=richard`. Caso contrário, o acesso será negado a ele. A chave de tag `Owner` corresponde a `Owner` e `owner` porque os nomes da chave de condição não fazem distinção entre maiúsculas e minúsculas. Para obter mais informações, consulte Elementos de política JSON do IAM: Condition.

Este exemplo mostra como criar uma política baseada em identidade que use a etiqueta `team` de entidade principal no ARN do recurso. A política concede permissão para excluir filas do Amazon Simple Queue Service, mas somente se o nome da fila começar com o nome da equipe seguido de `-queue`. Por exemplo, `qa-queue` se `qa` for o nome da equipe para a etiqueta da entidade principal `team`.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2:111122223333:${aws:PrincipalTag/team}-
queue"
      }
}
```

Controlar o acesso durante solicitações do AWS

É possível usar condições em suas políticas do IAM para controlar quais pares de chave-valor da tag podem ser passados em uma solicitação que aplica tags a um recurso da AWS.

Este exemplo mostra como você pode criar uma política baseada em identidade que permita usar a ação `CreateTags` do Amazon EC2 para anexar tags a uma instância. É possível anexar tags somente se a tag contiver a chave `environment` e os valores `preprod` ou `production`. Se quiser, você poderá usar o modificador `ForAllValues` com a chave de condição `aws:TagKeys` para indicar que somente a chave `environment` é permitida na solicitação. Isso impede que os usuários incluam outras chaves, mesmo acidentalmente, ao usar `Environment` em vez de `environment`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
```

```
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Controlar o acesso com base em chaves de tag

É possível usar uma condição em suas políticas do IAM para controlar se as chaves de etiquetas específicas podem ser usadas em uma solicitação.

Ao usar políticas para controlar o acesso usando etiquetas, convém usar a chave de condição `aws:TagKeys`. Os serviços da AWS que oferecem suporte a etiquetas podem permitir que você crie vários nomes de chaves de etiqueta que diferem apenas em termos de maiúsculas ou minúsculas, como etiquetar uma instância do Amazon EC2 com `stack=production` e `Stack=test`. Os nomes das chaves não diferenciam maiúsculas de minúsculas em condições da política. Isso significa que, se você especificar `"aws:ResourceTag/TagKey1": "Value1"` no elemento de condição da política, a condição corresponderá a uma chave de tag de recurso chamada `TagKey1` ou `tagkey1`, mas não ambas. Para evitar etiquetas duplicadas com uma chave que varia de acordo com o caso, use a condição `aws:TagKeys` para definir as chaves de etiqueta que seus usuários podem aplicar ou use políticas de etiquetas, disponíveis no AWS Organizations. Para obter mais informações, consulte Políticas de tags no Guia do usuário do AWS Organizations.

Este exemplo mostra como você pode criar uma política baseada em identidade que permita criar e marcar um segredo do Secrets Manager, mas apenas com as chaves de tag `environment` ou `cost-center`. A condição `Null` garante que a condição seja avaliada como `false` se não houver tags na solicitação.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
```

```
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

Acesso a recursos entre contas no IAM

Para alguns serviços da AWS, você pode conceder acesso entre contas para os seus recursos usando o IAM. Para fazer isso, você pode anexar uma política de recurso diretamente ao recurso que você deseja compartilhar ou usar um perfil como um proxy.

Para compartilhar o recurso diretamente, ele deve ser compatível com políticas baseadas em recursos. Ao contrário de uma política baseada em identidade, uma política baseada em recursos especifica quem (qual entidade principal) pode acessar esse recurso.

Use um perfil como proxy quando quiser acessar recursos em outra conta que não sejam compatíveis com políticas baseadas em recursos.

Para obter detalhes sobre as diferenças entre esses tipos de política, consulte Políticas baseadas em identidade e em recurso.

> **Note**
>
> As funções do IAM e as políticas baseadas em recurso delegam o acesso entre contas em uma única partição. Por exemplo, você tem uma conta no Oeste dos EUA (Norte da Califórnia) na partição `aws` padrão. Você também tem uma conta na China na partição `aws-cn`. Você não pode usar uma política baseada em recursos na sua conta na China para permitir acesso a usuários na sua conta da AWS padrão.

Acesso entre contas usando perfis

Nem todos os serviços da AWS são compatíveis com políticas baseadas em recursos. Para esses serviços, você pode usar perfis do IAM entre contas para centralizar o gerenciamento de permissões ao fornecer acesso entre contas a vários serviços. Um perfil do IAM entre contas é um perfil do IAM que inclui uma política de confiança que permite que as entidades principais do IAM em outra

conta da AWS assumam o perfil. Simplificando, você pode criar um perfil em uma conta da AWS que delega permissões específicas para outra conta da AWS.

Para obter informações sobre como anexar uma política a uma identidade do IAM, consulte Gerenciar políticas do IAM.

> **Note**
>
> Quando uma entidade principal muda para um perfil para usar temporariamente as permissões do perfil, ela abre mão de suas permissões originais e assume as permissões atribuídas ao perfil que assumiu.

Vamos dar uma olhada no processo geral aplicado ao software de parceiro da APN que precisa acessar uma conta de cliente.

1. O cliente cria um perfil do IAM na própria conta com uma política que permite acessar os recursos do Amazon S3 que o parceiro da APN exige. Neste exemplo, o nome do perfil é `APNPartner`.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "s3:*",
               "Resource": [
                   "arn:aws:s3:::bucket-name"
               ]
           }
       ]
   }
   ```

2. Em seguida, o cliente especifica que o perfil pode ser assumido pela conta da AWS do parceiro fornecendo o ID da Conta da AWS do parceiro da APN na política de confiança do perfil `APNPartner`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. O cliente fornece o nome do recurso da Amazon (ARN) do perfil ao parceiro da APN. O ARN é o nome totalmente qualificado do perfil.

```
arn:aws:iam::Customer-Account-ID:role/APNPartner
```

> **Note**
>
> Recomendamos usar uma ID externa em situações multilocatárias. Para obter detalhes, consulte Acesso às Contas da AWS de propriedade de terceiros.

4. Quando o software do parceiro da APN precisa acessar a conta do cliente, o software chama a API AssumeRole no AWS Security Token Service com o ARN do perfil na conta do cliente. O STS retorna uma credencial da AWS temporária que permite que o software faça seu trabalho.

Para ver outro exemplo de concessão de acesso entre contas usando perfis, consulte Acesso a um usuário do IAM em outra Conta da AWS de sua propriedade. Você também pode seguir o Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM.

Acesso entre contas usando políticas baseadas em recursos

Quando uma conta acessa um recurso por meio de outra conta usando uma política baseada em recursos, a entidade principal ainda trabalha na conta confiável e não precisa abrir mão de suas permissões para receber as permissões do perfil. Em outras palavras, a entidade principal continua

a ter acesso aos recursos na conta confiável ao mesmo tempo em que tem acesso ao recurso na conta de confiança. Isso é útil para tarefas como cópia de informações de ou para o recurso compartilhado em outra conta.

As entidades de segurança que você pode especificar em uma política baseada em recurso incluem contas, usuários do IAM, entidades principais de usuários federados do AWS STS, entidades principais federadas SAML, entidades principais federadas OIDC, perfis do IAM, sessões de perfis assumidos ou serviços da AWS. Para obter mais informações, consulte Especificar uma entidade principal.

Para saber se as entidades principais de contas fora de sua zona de confiança (organização ou conta confiável) têm acesso para assumir seus perfis, consulte Identificar recursos compartilhados com uma entidade principal externa.

A lista a seguir inclui alguns dos serviços da AWS que oferecem suporte às políticas baseadas em recursos. Para obter uma lista completa do número crescente de serviços da AWS que compatíveis com a anexação de políticas de permissão aos recursos em vez de nos principais, consulte AWSServiços da que funcionam com o IAMe procure os serviços que têm Sim na coluna Baseadas em recursos.

- Buckets do Amazon S3: a política é anexada ao bucket, mas controla o acesso ao bucket e aos objetos nele. Para obter mais informações, consulte Bucket policies for Amazon S3 no Guia do usuário do Amazon Simple Storage Service. Em alguns casos, pode ser melhor usar funções para acesso entre contas ao Amazon S3. Para mais informações, consulte as demonstrações de exemplo no Guia do usuário do Amazon Simple Storage Service.
- Tópicos do Amazon Simple Notification Service (Amazon SNS) — Para obter mais informações, acesse Casos de exemplo para controle de acesso do Amazon SNS no Guia do desenvolvedor do Amazon Simple Notification Service.
- Filas do Amazon Simple Queue Service (Amazon SQS) – Para obter mais informações, consulteApêndice: A linguagem da política de acesso no Guia do desenvolvedor do Amazon Simple Queue Service.

Políticas baseadas em recursos para delegar permissões da AWS

Se um recurso conceder permissões às entidades de segurança em sua conta, você poderá delegar essas permissões a identidades do IAM específicas. As identidades são usuários, grupos de usuários ou funções na conta. Você delega permissões anexando uma política à identidade. Você pode conceder até o máximo de permissões permitidas pela conta proprietária do recurso.

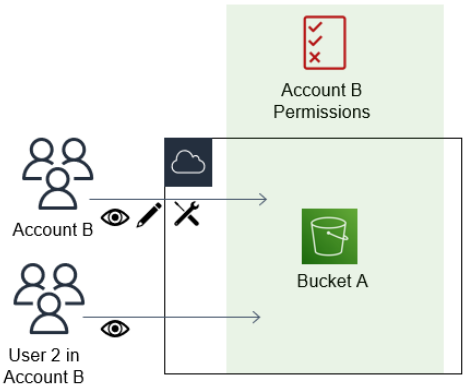
> **Important**
>
> No acesso entre contas, uma entidade principal precisa de uma `Allow` na política de identidade e uma política baseada em recursos.

Suponha que uma política baseada em recursos permita acesso administrativo total a um recurso a todos os principais na conta. Em seguida, você pode delegar acesso total, acesso somente leitura ou qualquer outro acesso parcial aos principais na sua conta da AWS. Como alternativa, se a política baseada em recursos permitir somente permissões de lista, você só poderá delegar o acesso à lista. Se você tentar delegar mais permissões do que sua conta possui, os principais ainda terão apenas acesso de lista.

Para obter mais informações sobre como essas decisões são tomadas, consulte Determinar se uma solicitação é permitida ou negada em uma conta.

> **Note**
>
> As funções do IAM e as políticas baseadas em recurso delegam o acesso entre contas em uma única partição. Por exemplo, não é possível adicionar acesso entre contas entre uma conta na partição padrão `aws` e uma conta na partição `aws-cn`.

Por exemplo, presuma que você gerencie a `AccountA` e a `AccountB`. Na AccountA, você tem o bucket do Amazon S3 chamado `BucketA`.



1. Anexe uma política baseada em recursos ao `BucketA` que permite que todas as entidades principais na AccountB tenham acesso total aos objetos no bucket. Eles podem criar, ler ou excluir objetos nesse bucket.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

A AccountA fornece acesso total à AccountB para o BucketA nomeando a AccountB como uma entidade principal na política baseada em recursos. Como resultado, a AccountB é autorizada a realizar qualquer ação no BucketA, e o administrador da AccountB pode delegar acesso aos seus usuários na AccountB.

O usuário raiz da AccountB tem todas as permissões concedidas à conta. Portanto, o usuário raiz tem acesso total ao BucketA.

2. Na AccountB, anexe uma política ao usuário do IAM denominado User2. Essa política permite que o usuário tenha acesso somente leitura aos objetos no BucketA. Isso significa que o User2 pode visualizar os objetos, mas não criá-los, editá-los ou excluí-los.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
```

```
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

O nível máximo de acesso que a AccountB pode delegar é o nível de acesso concedido à conta. Nesse caso, a política baseada em recursos concedeu acesso total à AccountB, mas o User2 tem acesso somente leitura.

O administrador da AccountB não concede acesso ao User1. Por padrão, os usuários não têm permissões, exceto aquelas explicitamente concedidas, então o User1 não tem acesso ao BucketA.

O IAM avalia as permissões de um principal no momento que este faz uma solicitação. Se você usar curingas (*) para fornecer aos usuários acesso total aos recursos, as entidades principais poderão acessar qualquer recurso ao qual a conta da AWS tenha acesso. Isso é verdadeiro mesmo para recursos que você adiciona ou aos quais obtém acesso após criar a política de usuário.

No exemplo anterior, se a AccountB tivesse anexado uma política ao User2 que permitisse acesso total aos recursos em todas as contas, o User2 teria acesso automaticamente a quaisquer recursos aos quais a AccountB tivesse acesso. Isso inclui o acesso ao BucketA e o acesso a quaisquer outros recursos concedido por políticas baseadas em recursos na AccountA.

Para obter mais informações sobre usos complexos de perfis, como conceder acesso a aplicações e serviços, consulte Cenários comuns para perfis do IAM.

> **Important**
>
> Conceda acesso somente a entidades confiáveis e atribua o nível mínimo de acesso necessário. Sempre que a entidade confiável for outra conta da AWS, qualquer entidade principal do IAM poderá ter acesso ao seu recurso. A conta confiável da AWS pode delegar acesso apenas na medida em que o acesso foi concedido; ela não pode delegar mais acessos do que o concedido para a própria conta.

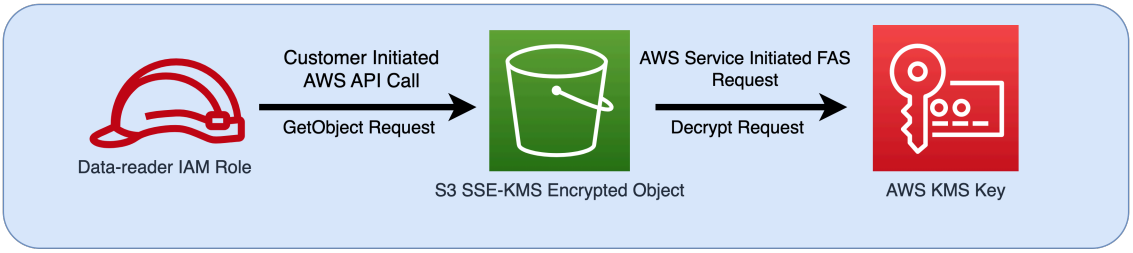
Para obter informações sobre permissões, políticas e a linguagem de política de permissões usada para criar políticas, consulte Gerenciamento de acesso para recursos da AWS.

Sessões de acesso direto

Sessões de acesso direto (FAS) é uma tecnologia do IAM usada pelos serviços da AWS para transmitir sua identidade, permissões e atributos de sessão quando um serviço da AWS faz uma solicitação em seu nome. O FAS usa as permissões da identidade que chama um serviço da AWS, combinadas com a identidade de um serviço da AWS, para fazer solicitações aos serviços subsequentes. As solicitações de FAS só são feitas aos serviços da AWS em nome de uma entidade principal do IAM depois que um serviço recebe uma solicitação que requeira interações com outros serviços ou recursos da AWS para ser atendida. Quando uma solicitação de FAS é feita:

- O serviço que recebe a solicitação inicial de uma entidade principal do IAM verifica as permissões da entidade principal do IAM.
- O serviço que recebe uma solicitação de FAS subsequente também verifica as permissões da mesma entidade principal do IAM.

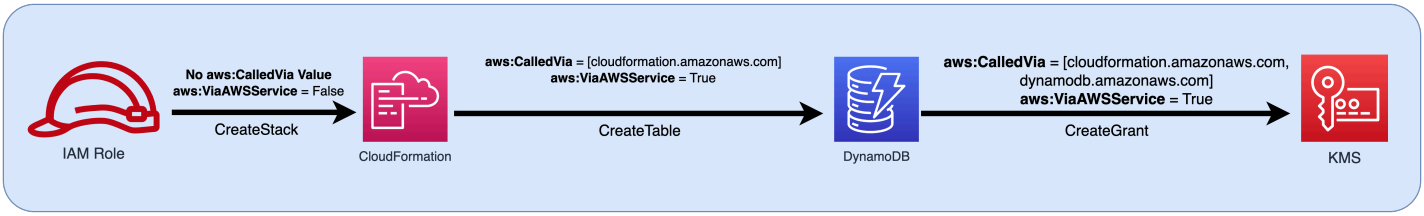
Por exemplo, o FAS é usado pelo Amazon S3 para fazer chamadas ao AWS Key Management Service para descriptografar um objeto quando o SSE-KMS foi usado para criptografá-lo. Ao baixar um objeto criptografado com o SSE-KMS, um perfil denominado data-reader chama getObject no objeto do Amazon S3, não chama o AWS KMS diretamente. Depois de receber a solicitação getObject e autorizar o data-reader, o Amazon S3 faz uma solicitação de FAS ao AWS KMS para descriptografar o objeto do Amazon S3. Quando o KMS recebe a solicitação de FAS, ele verifica as permissões do perfil e só autoriza a solicitação de descriptografia se o data-reader tem as permissões corretas na chave do KMS. As solicitações para o Amazon S3 e o AWS KMS são autorizadas usando as permissões do perfil, e elas só são bem-sucedidas se o leitor de dados tem permissões tanto para o objeto do Amazon S3 quanto para a chave do AWS KMS.



> **Note**
>
> Solicitações de FAS adicionais podem ser feitas pelos serviços que receberam uma solicitação de FAS. Nesses casos, a entidade principal solicitante deve ter permissões para todos os serviços chamados por FAS.

Solicitações de FAS e condições de política do IAM

Quando as solicitações de FAS são feitas, as chaves de condição aws:CalledVia, aws:CalledViaFirst e aws:CalledViaLast são preenchidas com a entidade principal do serviço que iniciou a chamada de FAS. O valor da chave de condição aws:ViaAWSService é definido como `true` sempre que uma solicitação de FAS é feita. No diagrama a seguir, a solicitação feita diretamente ao CloudFormation ainda não tem uma chave de condição `aws:CalledVia` ou `aws:ViaAWSService` configurada. Quando o CloudFormation e o DynamoDB fazem solicitações de FAS subsequentes em nome do perfil, os valores dessas chaves de condição são preenchidos.



Para permitir que uma solicitação de FAS seja feita, quando normalmente seria negada por uma instrução de política de negação com uma chave de condição testando os endereços IP ou as VPCs de origem, você deve usar chaves de condição para prever uma exceção para as solicitações de FAS em sua política de negação. Isso pode ser feito para todas as solicitações de FAS usando a chave de condição `aws:ViaAWSService`. Para permitir que apenas serviços específicos da AWS façam solicitações de FAS, use `aws:CalledVia`.

> **Important**
>
> Quando uma solicitação de FAS é feita após uma solicitação inicial ser ter sido feita por meio de um endpoint da VPC, os valores da chave de condição para `aws:SourceVpce`, `aws:SourceVpc` e `aws:VpcSourceIp` da solicitação inicial não são usados nas solicitações de FAS. Ao escrever políticas usando `aws:VpcSourceIp` ou `aws:SourceVpce` para conceder acesso condicionalmente, você também deve usar `aws:ViaAWSService` ou `aws:CalledVia` permitir solicitações de FAS. Quando uma solicitação de FAS for feita depois que uma solicitação inicial for recebida por um endpoint

público do serviço da AWS, as solicitações de FAS subsequentes serão feitas com o mesmo valor de chave de condição `aws:SourceIP`.

Exemplo: permitir acesso ao Amazon S3 a partir de uma VPC ou com FAS

No exemplo de política do IAM a seguir, as solicitações GetObject e as solicitações do Athena do Amazon S3 só serão permitidas se forem originadas de endpoints da VPC conectados a *example_vpc* ou se a solicitação for uma solicitação de FAS feita pelo Athena.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
          "vpc-111bbb22"
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
```

```
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Para obter exemplos adicionais do uso de chaves de condição para permitir acesso de FAS, consulte o repositório de exemplos de políticas de perímetro de dados.

Exemplos de políticas baseadas em identidade do IAM

Uma política é um objeto na AWS que, quando associado a uma identidade ou um recurso, define suas permissões. A AWS avalia essas políticas quando uma entidade de segurança do IAM (usuário ou função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas são armazenadas na AWS como documentos JSON que são anexados a uma identidade (usuário, grupo de usuários ou função) do IAM. As políticas baseadas em identidade incluem as políticas gerenciadas pela AWS, as políticas gerenciadas pelo cliente e as políticas em linha. Para saber como criar uma política do IAM usando esses exemplos de documentos de política JSON, consulte the section called “Criar políticas usando o editor de JSON”.

Por padrão, todas as solicitações são negadas, de modo que você deve fornecer acesso aos serviços, às ações e aos recursos que você pretende que a identidade acesse. Se você também quiser permitir acesso para concluir as ações especificadas no console do IAM, precisará fornecer permissões adicionais.

A seguinte biblioteca de políticas pode ajudar você a definir permissões para suas identidades do IAM. Depois de encontrar a política de que precisa, escolha view this policy (visualizar essa política) para visualizar o JSON da política. É possível usar o documento de política JSON como um modelo para suas próprias políticas.

> **Note**
>
> Se você quiser enviar uma política para ser incluída neste guia de referência, use o botão Feedback na parte inferior desta página.

Exemplos de políticas: AWS

- Permite acesso durante um intervalo específico de datas. (Visualizar essa política.)
- Permite habilitar e desabilitar as regiões da AWS. (Visualizar essa política.)
- Permite que os usuários autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança. (Visualizar essa política.)
- Permite acesso específico ao usar MFA durante um intervalo específico de datas. (Visualizar essa política.)
- Permite que os usuários gerenciem suas próprias credenciais na página Credenciais de segurança. (Visualizar essa política.)
- Permite que os usuários gerenciem seus próprios dispositivos de MFA na página Credenciais de segurança. (Visualizar essa política.)
- Permite que os usuários gerenciem suas próprias senhas na página Credenciais de segurança. (Visualizar essa política.)
- Permite que os usuários gerenciem suas próprias senhas, chaves de acesso e chaves públicas SSH na página Credenciais de segurança. (Visualizar essa política.)
- Nega o acesso à AWS com base na região solicitada. (Visualizar essa política.)
- Nega acesso à AWS com base no endereço IP da fonte. (Visualizar essa política.)

Exemplo de política: AWS Data Exchange

- Negar acesso aos recursos do Amazon S3 fora da conta, exceto o AWS Data Exchange. (Visualizar essa política.)

Exemplos de políticas: AWS Data Pipeline

- Nega acesso a pipelines não criados pelo usuário (Visualizar essa política.)

Exemplo de políticas: Amazon DynamoDB

- Permite acesso a uma tabela específica do Amazon DynamoDB (Visualize esta política.)
- Permite acesso aos atributos específicos do Amazon DynamoDB (Visualize esta política.)
- Permite acesso no nível do item ao Amazon DynamoDB com base em um ID do Amazon Cognito (Visualize esta política.)

Políticas de exemplo: Amazon EC2

- Permite anexar ou desvincular volumes do Amazon EBS para instâncias do Amazon EC2 com base em etiquetas (Visualize esta política.)
- Permite executar instâncias do Amazon EC2 em uma sub-rede específica, de forma programática e no console (Visualize esta política.)
- Permite gerenciar grupos de segurança do Amazon EC2 associados a uma VPC específica, de forma programática e no console (Visualize esta política.)
- Permite iniciar ou interromper instâncias do Amazon EC2 etiquetadas por um usuário, de forma programática e no console (Visualize esta política.)
- Permite iniciar ou interromper instâncias do Amazon EC2 com base em etiquetas de recurso e entidade de segurança, de forma programática e no console (Visualize esta política.)
- Permite iniciar ou interromper instâncias do Amazon EC2 quando as etiquetas de recurso e entidade de segurança coincidem (Visualize esta política.)
- Permite acesso total ao Amazon EC2 em uma região específica, de forma programática e no console. (Visualizar essa política.)
- Permite iniciar ou interromper uma instância específica do Amazon EC2 e modificar um determinado grupo de segurança, de forma programática e no console (Visualize esta política.)
- Nega acesso a operações do Amazon EC2 específicas sem MFA (Visualize esta política.)
- Limita o término de instâncias do Amazon EC2 a um intervalo de endereços IP específico (Visualize esta política.)

Exemplo de políticas do AWS Identity and Access Management (IAM)

- Permite acesso à API do simulador de políticas (Visualizar essa política.)
- Permite acesso ao console do simulador de políticas (Visualizar essa política.)
- Permite assumir funções que tenham uma tag específica, de forma programática e no console (Visualizar essa política).
- Permite e nega o acesso a vários serviços, de forma programática e no console (Visualizar essa política).
- Permite adicionar uma etiqueta específica a um usuário do IAM com uma etiqueta específica diferente, de forma programática e no console (Visualize esta política.)
- Permite adicionar uma etiqueta específica a qualquer usuário ou função do IAM, de forma programática e no console (Visualize esta política.)

- Permite criar um novo usuário somente com tags específicas (Visualizar essa política.)
- Permite gerar e recuperar os relatórios de credenciais do IAM (Visualize esta política.)
- Permite gerenciar uma associação do grupo, de forma programática e no console (Visualizar essa política).
- Permite gerenciar uma tag específica (Visualizar essa política.)
- Permite transmitir uma função do IAM para um serviço específico (Visualize esta política.)
- Permite o acesso somente leitura ao console do IAM sem gerar relatórios (Visualize esta política.)
- Permite o acesso somente leitura ao console do IAM (Visualize esta política.)
- Permite que usuários específicos gerenciem um grupo, de forma programática e no console (Visualizar essa política).
- Permite definir os requisitos de senha da conta, de forma programática e no console (Visualizar essa política).
- Permite usar a API do simulador de políticas para usuários com um caminho específico (Visualizar essa política.)
- Permite usar o console do simulador de políticas para usuários com um caminho específico (Visualizar essa política.)
- Permite que os usuários do IAM autogerenciem um dispositivo com MFA. (Visualizar essa política.)
- Permite que usuários do IAM definam as próprias credenciais de forma programática e no console. (Visualizar essa política.)
- Permite a exibição das informações de último acesso do serviço para uma política do AWS Organizations no console do IAM. (Visualizar essa política.)
- Limita as políticas gerenciadas que podem ser aplicadas a um usuário, grupo de usuários ou função do IAM (Visualize esta política.)
- Permite acesso às políticas do IAM somente na sua conta (Visualizar essa política.)

Exemplos de políticas: AWS Lambda

- Permite que uma função do AWS Lambda acesse uma tabela do Amazon DynamoDB (Visualize esta política.)

Políticas de exemplo: Amazon RDS

- Permite acesso total ao banco de dados do Amazon RDS em uma região específica. (Visualizar essa política.)
- Permite a restauração de bancos de dados do Amazon RDS de forma programática e no console (Visualize esta política)
- Permite aos proprietários de etiquetas o acesso total aos recursos do Amazon RDS que eles etiquetaram (Visualize esta política.)

Políticas de exemplo: Amazon S3

- Permite que um usuário do Amazon Cognito acesse objetos no seu próprio bucket do Amazon S3 (Visualize esta política.)
- Permite que um usuário com credenciais temporárias acesse seu próprio diretório base do Amazon S3, de forma programática e no console (Visualizar esta política).
- Permite acesso total ao S3, mas nega explicitamente o acesso ao bucket de produção se o administrador não estiver conectado usando MFA nos últimos trinta minutos (Visualizar essa política).
- Permite que usuários do IAM acessem seu próprio diretório base do Amazon S3, de forma programática e no console (Visualize esta política.)
- Permite que um usuário gerencie um único bucket do Amazon S3 e nega todas as outras ações e recursos da AWS (Visualize esta política.)
- Permite acesso `Read` e `Write` a um bucket específico do Amazon S3 (Visualize esta política.)
- Permite acesso `Read` e `Write` a um bucket específico do Amazon S3 de forma programática e no console (Visualize esta política.)

AWS: Permite o acesso com base na data e hora

Este exemplo mostra como é possível criar uma política do IAM que permita acesso a ações com base em data e hora. Esta política restringe o acesso às ações que ocorrem entre 1.º de abril de 2020 e 30 de junho de 2020 (UTC). Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Para saber mais sobre como usar várias condições dentro do bloco `Condition` de uma política do IAM, consulte Vários valores em uma condição.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> Você não pode usar uma variável de política com o operador de condição Date. Para saber mais, consulte Elemento de condição

AWS: permite habilitar e desabilitar regiões da AWS

Este exemplo mostra como você pode criar uma política do IAM que permita que um administrador habilite e desabilite a região Ásia-Pacífico (Hong Kong) (ap-east-1). Esta política define permissões para acesso programático e do console. Essa configuração aparece na página Configurações de contas no Console de gerenciamento da AWS. Essa página inclui informações confidenciais em nível de conta que devem ser visualizadas e gerenciadas apenas por administradores de contas. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

> **Important**
>
> Você não pode habilitar ou desabilitar regiões habilitadas por padrão. Você só pode incluir regiões desabilitadas por padrão. Para obter mais informações, consulte Como gerenciar regiões da AWS no Referência geral da AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários do IAM autenticados por autenticação multifator (MFA) gerenciem suas próprias credenciais

na página Credenciais de segurança. Essa página do Console de gerenciamento da AWS exibe as informações de conta, como ID de conta e ID de usuário canônico. Os usuários também podem visualizar e editar as próprias senhas, chaves de acesso, dispositivos MFA e certificados X.509, chaves SSH e credenciais do Git. Esta política de exemplo inclui as permissões necessárias para visualizar e editar todas as informações na página. Ela também exige que o usuário seja configurados e autenticado usando MFA para que possa executar outras operações na AWS. Para permitir que os usuários gerenciem as próprias credenciais sem usar MFA, consulte AWS: permite que usuários do IAM gerenciem suas próprias credenciais na página Credenciais de segurança.

Para saber como os usuários podem acessar a página Credenciais de segurança, consulte Como os usuários do IAM alteram a própria senha (console).

> **Note**
>
> - Este exemplo de política não permite que os usuários redefinam uma senha ao fazer login no Console de gerenciamento da AWS pela primeira vez. Recomendamos que você não conceda permissões a novos usuários até que eles façam login. Para obter mais informações, consulte Como faço para criar usuários do IAM com segurança?. Isso também impede que os usuários com uma senha expirada redefinam sua senha ao fazerem login. É possível permitir isso adicionando `iam:ChangePassword` e `iam:GetAccountPasswordPolicy` à instrução `DenyAllExceptListedIfNoMFA`. Porém, não recomendamos isso porque permitir que os usuários alterem a senha sem MFA pode ser um risco de segurança.
> - Caso pretenda usar essa política para acesso programático, será necessário chamar `GetSessionToken` para autenticar com o MFA. Para obter mais informações, consulte Acesso seguro à API com a MFA.

O que essa política faz?

- A instrução `AllowViewAccountInfo` permite que o usuário visualize informações no nível da conta. Essas permissões devem estar nas suas respectivas instruções, pois não oferecem suporte ou não precisam especificar um ARN de recurso. Em vez disso, as permissões especificam `"Resource" : "*"`. Essa instrução inclui as seguintes ações que permitem que o usuário visualize informações específicas:
  - `GetAccountPasswordPolicy`: visualizar os requisitos de senha da conta ao mudar a própria senha de usuário do IAM.

  - `ListVirtualMFADevices`: visualize detalhes sobre um dispositivo com MFA virtual habilitado para o usuário.
- A instrução `AllowManageOwnPasswords` permite que o usuário altere a própria senha. Essa instrução também inclui a ação `GetUser`, que é necessária para visualizar a maioria das informações na página My security credentials (Minhas credenciais de segurança).
- A instrução `AllowManageOwnAccessKeys` permite que o usuário crie, atualize e exclua as próprias chaves de acesso. O usuário também pode recuperar informações sobre quando a chave de acesso foi usada pela última vez.
- A instrução `AllowManageOwnSigningCertificates` permite que o usuário carregue, atualize e exclua os próprios certificados de assinatura.
- A instrução `AllowManageOwnSSHPublicKeys` permite que o usuário carregue, atualize e exclua as próprias chaves públicas SSH para o CodeCommit.
- A instrução `AllowManageOwnGitCredentials` permite que o usuário crie, atualize e exclua as próprias credenciais do Git para o CodeCommit.
- A instrução `AllowManageOwnVirtualMFADevice` permite que o usuário crie o próprio dispositivo de MFA virtual. O ARN do recurso dessa instrução permite que o usuário crie um dispositivo de MFA com qualquer nome, mas outras instruções da política permitem apenas que o usuário anexe o dispositivo ao usuário conectado.
- A instrução `AllowManageOwnUserMFA` permite que o usuário visualize ou gerencie o dispositivo MFA virtual, U2F ou de hardware para o próprio usuário. O recurso ARN nesta instrução permite acesso apenas ao próprio usuário do IAM do usuário. Os usuários não podem visualizar ou gerenciar o dispositivo MFA para outros usuários.
- A instrução `DenyAllExceptListedIfNoMFA` nega acesso a todas as ações em todos os serviços da AWS, exceto algumas ações listadas, mas somente se o usuário não estiver conectado com MFA. A instrução usa uma combinação de `"Deny"` e `"NotAction"` para negar explicitamente acesso a cada ação que não está listada. Os itens listados não são negados ou permitidos por essa instrução. No entanto, as ações são permitidas por outras instruções na política. Para obter mais informações sobre a lógica dessa instrução, consulte NotAction com Deny. Se o usuário está conectado com MFA, ocorrerá uma falha no teste de `Condition`, e a instrução não negará nenhuma ação. Neste caso, outras políticas ou instruções para o usuário determinam as permissões do usuário.

  Essa instrução garante que, quando o usuário não estiver conectado com MFA, ele só poderá executar as ações listadas. Além disso, eles poderão executar as ações listadas somente se outra instrução ou política permitir acesso a essas ações. Isso não permite que um usuário crie uma

senha no login, porque a ação `iam:ChangePassword` não deve ser permitida sem autorização de MFA.

A versão `...IfExists` do operador `Bool` garante que se a chave `aws:MultiFactorAuthPresent` estiver ausente, a condição retornará verdadeiro. Isso significa que, ao acessar uma API com credenciais de longo prazo, como uma chave de acesso, o usuário terá seu acesso negado às operações de API não relacionadas ao IAM.

Esta política não permite que os usuários visualizem a página Users (Usuários) no console do IAM ou usem essa página para acessar suas próprias informações de usuário. Para permitir que isso aconteça, adicione ação `iam:ListUsers` às instruções `AllowViewAccountInfo` e `DenyAllExceptListedIfNoMFA`. Ela também não permite que os usuários alterem a senha na própria página de usuário. Para permitir isso, adicione as ações `iam:GetLoginProfile` e `iam:UpdateLoginProfile` à instrução `AllowManageOwnPasswords`. Para permitir que um usuário altere a senha em sua própria página de usuário sem fazer login usando MFA, adicione a ação `iam:UpdateLoginProfile` à instrução `DenyAllExceptListedIfNoMFA`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
```

```
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
```

```
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
```

```
}
```

AWS: permite acesso específico com MFA em datas específicas

Este exemplo mostra como é possível criar uma política uma política baseada em identidade que use várias condições que são avaliadas usando um `AND` lógico. Ele permite acesso total ao serviço chamado `SERVICE-NAME-1` e acesso às ações `ACTION-NAME-A` e `ACTION-NAME-B` no serviço chamado `SERVICE-NAME-2`. Essas ações são permitidas somente quando o usuário é autenticado por meio da Autenticação multifator (MFA). O acesso é restrito a ações que ocorrem entre 1 de julho e 31 de dezembro 2017 2017 (UTC), inclusive. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Para saber mais sobre como usar várias condições dentro do bloco `Condition` de uma política do IAM, consulte Vários valores em uma condição.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS: permite que usuários do IAM gerenciem suas próprias credenciais na página Credenciais de segurança

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários do IAM gerenciem todas as suas próprias credenciais na página Credenciais de segurança. Essa página do Console de gerenciamento da AWS exibe as informações de conta, como ID de conta e ID de usuário canônico. Os usuários também podem visualizar e editar as próprias senhas, chaves de acesso, certificados X.509, chaves SSH e credenciais do Git. Esta política de exemplo inclui as permissões necessárias para visualizar e editar todas as informações na página, exceto o dispositivo MFA do usuário. Para permitir que os usuários gerenciem as próprias credenciais com MFA, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.

Para saber como os usuários podem acessar a página Credenciais de segurança, consulte Como os usuários do IAM alteram a própria senha (console).

O que essa política faz?

- A instrução `AllowViewAccountInfo` permite que o usuário visualize informações no nível da conta. Essas permissões devem estar nas suas respectivas instruções, pois não oferecem suporte ou não precisam especificar um ARN de recurso. Em vez disso, as permissões especificam `"Resource" : "*"`. Essa instrução inclui as seguintes ações que permitem que o usuário visualize informações específicas:
  - `GetAccountPasswordPolicy`: visualizar os requisitos de senha da conta ao mudar a própria senha de usuário do IAM.
  - `GetAccountSummary`: visualizar o ID da conta e o ID de usuário canônico da conta.
- A instrução `AllowManageOwnPasswords` permite que o usuário altere a própria senha. Essa instrução também inclui a ação `GetUser`, que é necessária para visualizar a maioria das informações na página My security credentials (Minhas credenciais de segurança).
- A instrução `AllowManageOwnAccessKeys` permite que o usuário crie, atualize e exclua as próprias chaves de acesso. O usuário também pode recuperar informações sobre quando a chave de acesso foi usada pela última vez.
- A instrução `AllowManageOwnSigningCertificates` permite que o usuário carregue, atualize e exclua os próprios certificados de assinatura.
- A instrução `AllowManageOwnSSHPublicKeys` permite que o usuário carregue, atualize e exclua as próprias chaves públicas SSH para o CodeCommit.

- A instrução `AllowManageOwnGitCredentials` permite que o usuário crie, atualize e exclua as próprias credenciais do Git para o CodeCommit.

Essa política não permite que os usuários visualizem ou gerenciem os próprios dispositivos MFA. Eles também não podem visualizar a página Users (Usuários) no console do IAM ou usar essa página para acessar as próprias informações de usuário. Para permitir isso, adicione a ação `iam:ListUsers` à instrução `AllowViewAccountInfo`. Ela também não permite que os usuários alterem a senha na própria página de usuário. Para permitir isso, adicione as ações `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile` e `iam:UpdateLoginProfile` à instrução `AllowManageOwnPasswords`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
```

```
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: permite que os usuários do IAM autenticados por MFA gerenciem seus próprios dispositivos de MFA na página Credenciais de segurança

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários do IAM autenticados por autenticação multifator (MFA) gerenciem seus próprios dispositivos de MFA na página Credenciais de segurança. Essa página do Console de gerenciamento da AWS exibe as informações de conta e usuário, mas o usuário só pode visualizar e editar o próprio dispositivo MFA. Para permitir que os usuários gerenciem todas as próprias credenciais com MFA, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.

> **Note**
>
> Se um usuário do IAM com essa política não for autenticado por MFA, essa política negará o acesso a todas as ações da AWS, exceto aquelas necessárias para autenticar usando MFA. Para usar a AWS CLI e a API da AWS, os usuários do IAM devem primeiro recuperar o token de MFA usando a operação GetSessionToken do AWS STS e usar esse token para autenticar a operação desejada. Outras políticas, como políticas baseadas em recurso ou outras políticas baseadas em identidade, podem permitir ações em outros serviços. Esta política negará esse acesso se o usuário do IAM não for autenticado por MFA.

Para saber como os usuários podem acessar a página Credenciais de segurança, consulte Como os usuários do IAM alteram a própria senha (console).

O que essa política faz?

- A instrução `AllowViewAccountInfo` permite que o usuário visualize detalhes sobre um dispositivo MFA virtual que está habilitado para ele. Essa permissão deve estar em sua própria instrução, pois ele não oferece suporte para especificar um ARN de recurso. Em vez disso, você deve especificar `"Resource" : "*"`.
- A instrução `AllowManageOwnVirtualMFADevice` permite que o usuário crie o próprio dispositivo de MFA virtual. O ARN do recurso dessa instrução permite que o usuário crie um dispositivo de MFA com qualquer nome, mas outras instruções da política permitem apenas que o usuário anexe o dispositivo ao usuário conectado.
- A instrução `AllowManageOwnUserMFA` permite que o usuário visualize ou gerencie o próprio dispositivo MFA virtual, U2F ou de hardware. O recurso ARN nesta instrução permite acesso

apenas ao próprio usuário do IAM do usuário. Os usuários não podem visualizar ou gerenciar o dispositivo MFA para outros usuários.

- A instrução `DenyAllExceptListedIfNoMFA` nega acesso a todas as ações em todos os serviços da AWS, exceto algumas ações listadas, mas somente se o usuário não estiver conectado com MFA. A instrução usa uma combinação de `"Deny"` e `"NotAction"` para negar explicitamente acesso a cada ação que não está listada. Os itens listados não são negados ou permitidos por essa instrução. No entanto, as ações são permitidas por outras instruções na política. Para obter mais informações sobre a lógica dessa instrução, consulte NotAction com Deny. Se o usuário está conectado com MFA, ocorrerá uma falha no teste de `Condition`, e a instrução não negará nenhuma ação. Neste caso, outras políticas ou instruções para o usuário determinam as permissões do usuário.

  Essa instrução garante que quando o usuário não fizer login com MFA, ele só poderá executar as ações listadas. Além disso, eles poderão executar as ações listadas somente se outra instrução ou política permitir acesso a essas ações.

  A versão `...IfExists` do operador `Bool` garante que se a chave `aws:MultiFactorAuthPresent` estiver ausente, a condição retornará verdadeiro. Isso significa que, ao acessar uma operação de API com credenciais de longo prazo, como uma chave de acesso, o usuário terá seu acesso negado às operações de API não relacionadas ao IAM.

Esta política não permite que os usuários visualizem a página Users (Usuários) no console do IAM ou usem essa página para acessar suas próprias informações de usuário. Para permitir que isso aconteça, adicione ação `iam:ListUsers` às instruções `AllowViewAccountInfo` e `DenyAllExceptListedIfNoMFA`.

> **Warning**
>
> Não adicione permissão para excluir um dispositivo MFA sem autenticação MFA. Os usuários com essa política podem tentar atribuir um dispositivo MFA virtual a si mesmos e receber um erro informando que não estão autorizados a executar `iam:DeleteVirtualMFADevice`. Se isso acontecer, não adicione essa permissão à instrução `DenyAllExceptListedIfNoMFA`. Os usuários que não são autenticados usando MFA nunca devem ter permissão para excluir o dispositivo MFA. Os usuários poderão ver esse erro se tiverem começado a atribuir um dispositivo MFA virtual a seu usuário anteriormente e cancelaram o processo. Para resolver esse problema, você ou outro administrador deve excluir o dispositivo MFA virtual existente do usuário usando a AWS CLI

ou a API da AWS. Para obter mais informações, consulte Não estou autorizado a executar: iam:DeleteVirtualMFADevice.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
```

```
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS: permite que os usuários do IAM alterem suas próprias senhas do console na página Credenciais de segurança

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários do IAM alterem suas próprias senhas do Console de gerenciamento da AWS na página Credenciais de segurança. Essa página do Console de gerenciamento da AWS exibe as informações de conta e usuário, mas o usuário só pode acessar a própria senha. Para permitir que os usuários gerenciem todas as próprias credenciais com MFA, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança. Para permitir que os usuários gerenciem as próprias credenciais sem usar MFA, consulte AWS: permite que usuários do IAM gerenciem suas próprias credenciais na página Credenciais de segurança.

Para saber como os usuários podem acessar a página Credenciais de segurança, consulte Como os usuários do IAM alteram a própria senha (console).

O que essa política faz?

- A instrução `ViewAccountPasswordRequirements` permite que o usuário visualize os requisitos de senha da conta ao mudar a própria senha de usuário do IAM.
- A instrução `ChangeOwnPassword` permite que o usuário altere a própria senha. Essa instrução também inclui a ação `GetUser`, que é necessária para visualizar a maioria das informações na página My security credentials (Minhas credenciais de segurança).

Esta política não permite que os usuários visualizem a página Users (Usuários) no console do IAM ou usem essa página para acessar suas próprias informações de usuário. Para permitir isso, adicione a ação `iam:ListUsers` à instrução `ViewAccountPasswordRequirements`. Ela também não permite que os usuários alterem a senha na própria página de usuário. Para permitir isso, adicione as ações `iam:GetLoginProfile` e `iam:UpdateLoginProfile` à instrução `ChangeOwnPassword`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: permite que os usuários do IAM gerenciem suas próprias senhas, chaves de acesso e chaves públicas SSH na página Credenciais de segurança

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários do IAM gerenciem suas próprias senhas, chaves de acesso e certificados X.509 na página Credenciais de segurança. Essa página do Console de gerenciamento da AWS exibe as informações de conta, como ID de conta e ID de usuário canônico. Os usuários também podem visualizar e editar as próprias senhas, chaves de acesso, dispositivos MFA, certificados X.509, chaves SSH e credenciais do Git. Esta política de exemplo inclui as permissões que são necessárias

para visualizar e editar somente a senha, chaves de acesso e certificado X.509. Para permitir que os usuários gerenciem todas as próprias credenciais com MFA, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança. Para permitir que os usuários gerenciem as próprias credenciais sem usar MFA, consulte AWS: permite que usuários do IAM gerenciem suas próprias credenciais na página Credenciais de segurança.

Para saber como os usuários podem acessar a página Credenciais de segurança, consulte Como os usuários do IAM alteram a própria senha (console).

O que essa política faz?

- A instrução `AllowViewAccountInfo` permite que o usuário visualize informações no nível da conta. Essas permissões devem estar nas suas respectivas instruções, pois não oferecem suporte ou não precisam especificar um ARN de recurso. Em vez disso, as permissões especificam `"Resource" : "*"`. Essa instrução inclui as seguintes ações que permitem que o usuário visualize informações específicas:
  - `GetAccountPasswordPolicy`: visualizar os requisitos de senha da conta ao mudar a própria senha de usuário do IAM.
  - `GetAccountSummary`: visualizar o ID da conta e o ID de usuário canônico da conta.
- A instrução `AllowManageOwnPasswords` permite que o usuário altere a própria senha. Essa instrução também inclui a ação `GetUser`, que é necessária para visualizar a maioria das informações na página My security credentials (Minhas credenciais de segurança).
- A instrução `AllowManageOwnAccessKeys` permite que o usuário crie, atualize e exclua as próprias chaves de acesso. O usuário também pode recuperar informações sobre quando a chave de acesso foi usada pela última vez.
- A instrução `AllowManageOwnSSHPublicKeys` permite que o usuário carregue, atualize e exclua as próprias chaves públicas SSH para o CodeCommit.

Essa política não permite que os usuários visualizem ou gerenciem os próprios dispositivos MFA. Eles também não podem visualizar a página Users (Usuários) no console do IAM ou usar essa página para acessar as próprias informações de usuário. Para permitir isso, adicione a ação `iam:ListUsers` à instrução `AllowViewAccountInfo`. Ela também não permite que os usuários alterem a senha na própria página de usuário. Para permitir isso, adicione as ações `iam:GetLoginProfile` e `iam:UpdateLoginProfile` à instrução `AllowManageOwnPasswords`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
```

```
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: nega acesso à AWS com base na região solicitada

Este exemplo mostra como é possível criar uma política baseada em identidade que negue acesso a todas as ações fora das regiões especificadas usando a chave de condição `aws:RequestedRegion`, com exceção das ações nos serviços especificados usando `NotAction`. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Essa política usa o elemento `NotAction` com o efeito `Deny`, que nega explicitamente o acesso a todas as ações não listadas na declaração. Ações nos serviços CloudFront, IAM, Route 53 e Suporte não devem ser negadas porque são produtos globais populares da AWS com um único endpoint fisicamente localizado na região `us-east-1`. Como todas as solicitações para esses serviços são feitas para a região `us-east-1`, as solicitações seriam negadas sem o elemento `NotAction`. Edite esse elemento para incluir ações para outros serviços globais da AWS que você usa, como `budgets`, `globalaccelerator`, `importexport`, `organizations` ou `waf`. Alguns outros serviços globais, como o Amazon Q Developer em aplicações de chat e o AWS Device Farm, são serviços globais com endpoints localizados fisicamente na região `us-west-2`. Para saber mais sobre todos os serviços que têm um único endpoint global, consulte Regiões e endpoints da AWS na Referência geral da AWS. Para obter mais informações sobre como usar o elemento `NotAction` com o efeito `Deny`, consulte Elementos de política JSON do IAM: NotAction.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "organizations:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

AWS: nega acesso à AWS com base no IP da fonte

Este exemplo mostra como você pode criar uma política do IAM que negue acesso a todas as ações da AWS na conta quando a solicitação vem de entidades principais fora do intervalo de IP especificado. A política é útil quando os endereços IP para sua empresa estão dentro dos intervalos especificados. Neste exemplo, a solicitação será negada, a menos que seja originária da faixa de CIDR 192.0.2.0/24 ou 203.0.113.0/24. A política não nega solicitações feitas pelos serviços da AWS usando Sessões de acesso direto pois o endereço IP do solicitante original é mantido.

Tenha cuidado ao usar condições negativas na mesma declaração de política que `"Effect": "Deny"`. Ao fazer isso, as ações especificadas na declaração de política são explicitamente negadas em todas as condições, exceto as especificadas.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

Quando outras políticas permitem ações, os principais podem fazer solicitações dentro do intervalo de endereços IP. Um serviço da AWS também pode fazer solicitações usando as credenciais do principal. Quando um principal faz uma solicitação fora do intervalo de IP, a solicitação é negada.

Para obter mais informações sobre o uso da chave de condição `aws:SourceIp`, incluindo informações sobre quando `aws:SourceIp` pode não funcionar na política, consulte AWSChaves de contexto de condições globais da.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: negar acesso aos recursos do Amazon S3 fora da sua conta, exceto o AWS Data Exchange

Este exemplo mostra como é possível criar uma política baseada em identidade que negue acesso a todos os recursos na AWS que não pertençam à conta, exceto os recursos que o AWS Data Exchange exigir para as operações normais. Para usar esta política, substitua o *texto do espaço*

reservado em itálico na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Você pode criar uma política semelhante para restringir o acesso aos recursos dentro de uma organização ou unidade organizacional, levando em conta os recursos pertencentes ao AWS Data Exchange, usando as chaves de condição `aws:ResourceOrgPaths` e `aws:ResourceOrgID`.

Se você usar AWS Data Exchange no ambiente, o serviço cria e interage com recursos como buckets do Amazon S3 pertencentes à conta de serviço. Por exemplo, o AWS Data Exchange envia solicitações aos buckets do Amazon S3 de propriedade do serviço AWS Data Exchange em nome da entidade principal do IAM (usuário ou perfil) que invoca as APIs do AWS Data Exchange. Nesse caso, o uso de `aws:ResourceAccount`, `aws:ResourceOrgPaths` ou `aws:ResourceOrgID` em uma política, sem levar em conta os recursos pertencentes ao AWS Data Exchange, negará acesso aos buckets pertencentes à conta do serviço.

- A instrução `DenyAllAwsResourcesOutsideAccountExceptS3` usa o elemento `NotAction` com o efeito Deny (Negar) que nega explicitamente o acesso a todas as ações não listadas na instrução e que também não pertencem à conta listada. O elemento `NotAction` indica as exceções a essa instrução. Essas ações são a exceção à essa instrução porque se as ações forem realizadas em recursos criados pelo AWS Data Exchange, a política as negará.
- A instrução `DenyAllS3ResourcesOutsideAccountExceptDataExchange` usa uma combinação das chaves de condições `ResourceAccount` e `CalledVia` para negar acesso às três ações do Amazon S3 excluídas na instrução anterior. A instrução nega as ações se os recursos não pertencerem à conta listada e se o serviço que está chamando não for o AWS Data Exchange. Essa instrução não nega as ações se o recurso pertencer à conta listada ou se a entidade principal do serviço listado, `dataexchange.amazonaws.com`, realizar as operações.

> **Important**
>
> Esta política não permite qualquer ação. Ela usa o efeito `Deny`, que nega explicitamente o acesso a todas as ações não listadas na instrução que não pertencerem à conta listada. Use essa política em combinação com outras políticas que permitem acesso a recursos específicos.

O exemplo a seguir mostra como você pode configurar a política para permitir acesso aos buckets do Amazon S3 necessários.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
```

```
    }
  ]
}
```

AWS Data Pipeline: nega acesso a pipelines do DataPipeline que não foram criados por um usuário

Este exemplo mostra como é possível criar uma política baseada em identidade que negue acesso aos pipelines que não foram criados por um usuário. Se o valor do campo `PipelineCreator` corresponder ao nome do usuário do IAM, as ações especificadas não serão negadas. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
```

```
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator":
 "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB: permite acesso a uma tabela específica

Este exemplo mostra como você pode criar uma política baseada em identidade que permita total acesso à tabela `MyTable` do DynamoDB. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

> **Important**
>
> Esta política permite todas as ações que podem ser executadas em uma tabela do DynamoDB. Para revisar essas ações, consulte Permissões da API do DynamoDB API: referência de ações, recursos e condições no Guia do desenvolvedor do Amazon DynamoDB. Você pode fornecer as mesmas permissões listando cada uma das ações. No entanto, se você usar o curinga (`*`) no elemento `Action`, como `"dynamodb:List*"`, não será necessário atualizar sua política se o DynamoDB adicionar uma nova ação Listar.

Esta política permite ações apenas em tabelas do DynamoDB que existem com o nome especificado. Para permitir que seus usuários tenham acesso `Read` a tudo no DynamoDB, você também pode anexar a política gerenciada pela AWS AmazonDynamoDBReadOnlyAccess.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB: permite acesso a atributos específicos

Este exemplo mostra como você pode criar uma política baseada em identidade que permita o acesso aos atributos específicos do DynamoDB. Esta política concede as permissões necessárias

para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

O requisito `dynamodb:Select` impede que a ação da API retorne qualquer atributo que não seja permitido, tal como de uma projeção de índice. Para saber mais sobre chaves de condição do DynamoDB, consulte Especificação de condições: uso de chaves de condição no Guia do desenvolvedor do Amazon DynamoDB. Para saber mais sobre como usar várias condições ou várias chaves de condições dentro do bloco `Condition` de uma política do IAM, consulte Vários valores em uma condição.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select":
 "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
```

```
}
```

O Amazon DynamoDB: permite acesso no nível do item ao DynamoDB com base em um ID do Amazon Cognito

Este exemplo mostra como é possível criar uma política baseada em identidade que permita acesso em nível de item à tabela `MyTable` do DynamoDB com base em um ID de usuário do banco de identidades do Amazon Cognito. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Para usar essa política, você deve estruturar sua tabela do DynamoDB para que o ID de usuário do banco de identidades do Amazon Cognito seja a chave de partição. Para obter mais informações, consulte CRiar uma tabela no Guia do desenevolvedor do Amazon DynamoDB.

Para saber mais sobre chaves de condição do DynamoDB, consulte Especificação de condições: uso de chaves de condição no Guia do desenvolvedor do Amazon DynamoDB.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-
identity.amazonaws.com:sub}"]
                }
            }
```

```
        }
    ]
}
```

Amazon EC2: anexar ou desvincular volumes do Amazon EBS para instâncias do EC2 com base em etiquetas

Este exemplo mostra como você pode criar uma política baseada em identidade que permita aos proprietários de volume do EBS anexar ou desanexar os volumes do EBS definidos usando a tag `VolumeUser` para instâncias do EC2 marcadas como instâncias de desenvolvimento (`Department=Development`). Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Para obter mais informações sobre a criação de políticas do IAM para controlar o acesso aos recursos do Amazon EC2, consulte Controle de acesso aos recursos do Amazon EC2 no Guia do usuário do Amazon EC2.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
```

```
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2: permite iniciar instâncias do EC2 em uma sub-rede específica, de forma programática e no console

Este exemplo mostra como você pode criar um política baseada em identidade que permita listar informações para todos os objetos do EC2 e executar instâncias do EC2 em uma sub-rede específica. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
```

```
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2: permite gerenciar grupos de segurança do EC2 com uma etiqueta especifica de par chave-valor de maneira programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que conceda aos usuários permissão para executar determinadas ações para grupos de segurança que têm a mesma tag. Esta política concede permissão para visualizar grupos de segurança no console do Amazon EC2, adicionar e remover regras de entrada e de saída, bem como listar e modificar descrições de regras para grupos de segurança existentes com a tag `Department=Test`. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:ModifySecurityGroupRules",
```

```
            "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
            "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
         ],
         "Resource": [
            "arn:aws:ec2:us-east-1:111122223333:security-group/*"
         ],
         "Condition": {
            "StringEquals": {
               "aws:ResourceTag/Department": "Test"
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:ModifySecurityGroupRules"
         ],
         "Resource": [
            "arn:aws:ec2:us-east-1:111122223333:security-group-rule/*"
         ]
      }
   ]
}
```

Amazon EC2: permite iniciar ou interromper instâncias do EC2 que um usuário etiquetou, de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que um usuário do IAM inicie ou interrompa instâncias do EC2, mas apenas se a tag `Owner` da instância tiver o valor do nome de usuário desse usuário. Esta política define permissões para acesso programático e do console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
```

```
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2: iniciar ou interromper instâncias baseadas em etiquetas

Este exemplo mostra como você pode criar uma política baseada em identidade que permita iniciar ou interromper instâncias com o par de chave-valor da etiqueta `Project = DataAnalytics`, mas apenas por entidades de segurança com o par de chave-valor da etiqueta `Department = Data`. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A condição na política retorna verdadeiro se ambas as partes da condição forem verdadeiras. A instância deve ter a tag `Project=DataAnalytics`. Além disso, a entidade de segurança (usuário ou função) do IAM que faz a solicitação deve ter a etiqueta `Department=Data`.

> **Note**
>
> Como prática recomendada, anexe políticas com a chave de condição `aws:PrincipalTag` aos grupos do IAM, para o caso de alguns usuários possuírem a etiqueta especificada e outros não.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2: iniciar ou interromper instâncias baseadas em etiquetas de entidade de segurança e recurso correspondentes

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que uma entidade principal inicie ou interrompa uma instância do Amazon EC2 quando a tag do recurso da instância e a tag da entidade de serviço têm o mesmo valor para a chave de tag `CostCenter`. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

> **Note**
>
> Como prática recomendada, anexe políticas com a chave de condição `aws:PrincipalTag` aos grupos do IAM, para o caso de alguns usuários possuírem a etiqueta especificada e outros não.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2: permite acesso total ao EC2 dentro de uma região específica, de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita acesso total ao EC2 em uma região específica. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política. Para obter uma lista de códigos de região, consulte Regiões disponíveis no Guia do usuário do Amazon EC2.

Como alternativa, você pode usar a chave de condição global, `aws:RequestedRegion`, que é compatível com todas as ações da API do Amazon EC2. Para obter mais informações, consulte Exemplo: restrição de acesso a uma região específica) no Guia do usuário do Amazon EC2.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2: permite iniciar ou interromper uma instância do EC2 e modificar um grupo de segurança de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita iniciar ou interromper uma instância específica do EC2 e modificar um grupo de segurança específico. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
```

```
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Amazon EC2: exige MFA (GetSessionToken) para operações específicas do EC2

Este exemplo mostra como você pode criar uma política baseada em identidade que permita acesso total a todas as operações de API da AWS no Amazon EC2. No entanto, ela negará explicitamente o acesso às operações de API `TerminateInstances` e `StopInstances` se o usuário não estiver autenticado usando Multi-Factor Authentication (MFA). Para fazer isso de forma programática, o usuário deve incluir os valores opcionais `SerialNumber` e `TokenCode` ao chamar a operação `GetSessionToken`. Essa operação retorna credenciais temporárias que foram autenticadas usando MFA. Para saber mais sobre GetSessionToken, consulte Solicitar credenciais para usuários em ambientes não confiáveis.

O que essa política faz?

- A instrução `AllowAllActionsForEC2` permite todas as ações do Amazon EC2.
- A instrução `DenyStopAndTerminateWhenMFAIsNotPresent` nega as ações `StopInstances` e `TerminateInstances` quando o contexto da MFA está ausente. Isso significa que as ações são negadas quando o contexto da autenticação multifator está ausente (o que significa que a MFA não foi usada). Uma negação substitui a permissão.

> **Note**
>
> A verificação da condição de `MultiFactorAuthPresent` na instrução `Deny` não deve ser `{"Bool":{"aws:MultiFactorAuthPresent":false}}` pois essa chave não está presente e não pode ser avaliada quando a MFA não é usada. Em vez disso, use a verificação `BoolIfExists` para ver se a chave está presente antes de verificar o valor. Para obter mais informações, consulte Operadores de condição ...IfExists.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2: limita o término de instâncias do EC2 para um intervalo de endereços IP

Este exemplo mostra como você pode criar uma política baseada em identidade que limite as instâncias do EC2 permitindo a ação, mas negando explicitamente o acesso quando a solicitação

vier de fora do intervalo de IP especificado. A política é útil quando os endereços IP para sua empresa estão dentro dos intervalos especificados. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Se essa política for usada em combinação com outras políticas que permitem a ação `ec2:TerminateInstances` (tal como a política gerenciada pela AWS AmazonEC2FullAccess), o acesso será negado. Isso ocorre porque uma instrução de negação explícita tem precedência sobre instruções para permitir. Para obter mais informações, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso.

> **Important**
>
> A chave de condição `aws:SourceIp` nega acesso a um AWS service (Serviço da AWS), tal como o AWS CloudFormation, que realiza chamadas em seu nome. Para mais informações sobre o uso da chave de condição `aws:SourceIp`, consulte AWSChaves de contexto de condições globais da.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
```

```
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM: acessar a API do simulador de política

Este exemplo mostra como você pode criar uma política baseada em identidade que permita usar a API do simulador de políticas para políticas anexadas a um usuário, grupo ou perfil na Conta da AWS atual. Esta política também permite acesso para simular políticas menos confidenciais transmitidas para a API como strings. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Para permitir que um usuário acesse o console do simulador de políticas para simular políticas anexadas a um usuário, grupo ou perfil na Conta da AWS atual, consulte IAM: acessar o console do simulador de política.

IAM: acessar o console do simulador de política

Este exemplo mostra como você pode criar uma política baseada em identidade que permita usar o console do simulador de políticas para políticas anexadas a um usuário, grupo ou perfil na atual Conta da AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "access-analyzer:ValidatePolicy",
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListPolicies",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM: assumir funções que têm uma etiqueta específica

Este exemplo mostra como você pode criar um política baseada em identidade que permita que um usuário do IAM assuma perfis com o par de chave-valor de etiqueta `Project = ExampleCorpABC`. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Se existir uma função com essa tag na mesma conta do usuário, o usuário poderá assumir essa função. Se uma função com essa tag existir em uma outra conta que não a do usuário, ela exigirá permissões adicionais. A política de confiança da função entre contas também deve permitir que o usuário ou todos os membros da conta do usuário assumam a função. Para obter informações sobre como usar funções para acesso entre contas, consulte Acesso a um usuário do IAM em outra Conta da AWS de sua propriedade.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM: permite e nega acesso a vários serviços de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita acesso total a vários serviços e acesso limitado autogerenciado no IAM. Ela também nega acesso ao bucket `logs` do Amazon S3 ou à instância `i-1234567890abcdef0` do Amazon EC2. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do`*

espaço reservado em itálico na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

> **Warning**
>
> Essa política permite acesso total a todas as ações e recursos em vários serviços. Essa política deve ser aplicada apenas a administradores confiáveis.

Você pode usar esta política como um limite de permissões para definir o número máximo de permissões que uma política baseada em identidade pode conceder a um usuário do IAM. Para obter mais informações, consulte Delegar responsabilidade para outras pessoas usando limites de permissões. Quando a política é usada como um limite de permissões para um usuário, as instruções definem os seguintes limites:

- Uma instrução `AllowServices` permite acesso total aos serviços especificados da AWS. Isso significa que as ações do usuário nesses serviços são limitadas apenas pelas políticas de permissões que são anexadas ao usuário.
- A instrução `AllowIAMConsoleForCredentials` permite acesso para listar todos os usuários do IAM. Esse acesso é necessário para navegar na página Usuários no Console de gerenciamento da AWS. Ela também permite visualizar os requisitos de senha da conta, que são necessários para o usuário alterar a própria senha.
- A instrução `AllowManageOwnPasswordAndAccessKeys` permite que os usuários gerenciem apenas suas próprias chaves de acesso programático e senha do console. Isso é importante porque se outra política conceder a um usuário acesso total ao IAM, esse usuário poderá alterar suas próprias permissões ou as permissões de outros usuários. Essa instrução impede que isso ocorra.
- A instrução `DenyS3Logs` nega explicitamente o acesso ao bucket de `logs`. Essa política impõe restrições da empresa ao usuário.
- A instrução `DenyEC2Production` nega explicitamente o acesso à instância de `i-1234567890abcdef0`.

Essa política não permite acesso a outros serviços ou ações. Quando a política é usada como um limite de permissões para um usuário, mesmo que outras políticas anexadas ao usuário permitam essas ações, o AWS nega a solicitação.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
```

```
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM: adicionar uma etiqueta específica a um usuário com uma etiqueta específica

Este exemplo mostra como você pode criar uma política baseada em identidade que permita adicionar a chave de tag `Department` com os valores de tag `Marketing`, `Development` ou `QualityAssurance` a um usuário do IAM. Esse usuário já deve incluir o par de chave-valor da etiqueta `JobFunction = Manager`. Você pode usar essa política para exigir que um gerente pertença apenas a um de três departamentos. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A instrução `ListTagsForAllUsers` permite a visualização de tags para todos os usuários em sua conta.

A primeira condição na instrução `TagManagerWithSpecificDepartment` usa o operador de condição `StringEquals`. A condição retorna verdadeiro se ambas as partes da condição forem verdadeiras. O usuário a ser marcado já deve ter a tag `JobFunction=Manager`. A solicitação deve incluir a chave de tag `Department` com um dos valores de tag listados.

A segunda condição usa o operador de condição `ForAllValues:StringEquals`. A condição retornará verdadeiro se todas as chaves de tag na solicitação corresponderem à chave na política. Isso significa que a única chave de tag na solicitação deve ser `Department`. Para obter mais informações sobre o uso de `ForAllValues`, consulte Operadores de conjunto para chaves de contexto de vários valores.

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM: adicionar uma etiqueta específica com valores específicos

Este exemplo mostra como você pode criar uma política baseada em identidade que permita adicionar apenas a chave de tag `CostCenter` e o valor de tag `A-123` ou o valor de tag `B-456` a qualquer usuário ou perfil do IAM. Use esta política para limitar a marcação a uma chave de tag específica e a um conjunto de valores de tag. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A instrução `ConsoleDisplay` permite a visualização de tags para todos os usuários e funções em sua conta.

A primeira condição na instrução `AddTag` usa o operador de condição `StringEquals`. A condição retorna verdadeiro se a solicitação inclui a chave de tag `CostCenter` com um dos valores de tag listados.

A segunda condição usa o operador de condição `ForAllValues:StringEquals`. A condição retornará verdadeiro se todas as chaves de tag na solicitação corresponderem à chave na política. Isso significa que a única chave de tag na solicitação deve ser `CostCenter`. Para obter mais informações sobre o uso de `ForAllValues`, consulte Operadores de conjunto para chaves de contexto de vários valores.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
```

```
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM: criar novos usuários somente com etiquetas específicas

Este exemplo mostra como você pode criar uma política baseada em identidade que permita a criação de usuários do IAM, mas apenas com uma ou ambas as chaves de tag `Department` e `JobFunction`. A chave de tag `Department` deve ter a chave de tag `Development` ou `QualityAssurance`. A chave de tag `JobFunction` deve ter o valor de tag `Employee`. Você pode usar essa política para exigir que novos usuários tenham um cargo e um departamento específicos. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A primeira condição na instrução usa o operador de condição `StringEqualsIfExists`. Se uma tag com a chave `Department` ou `JobFunction` estiver presente na solicitação, a tag deverá ter o valor especificado. Se nenhuma chave estiver presente, essa condição será avaliada como verdadeira. A única maneira de a condição ser avaliada como falsa é se uma das chaves de condição especificada estiver presente na solicitação, mas tiver um valor diferente dos permitidos. Para obter mais informações sobre o uso de `IfExists`, consulte Operadores de condição ...IfExists.

A segunda condição usa o operador de condição `ForAllValues:StringEquals`. A condição retorna verdadeira se houver uma correspondência entre cada um dos valores de chave especificados na solicitação e pelo menos um valor na política. Isso significa que todas as tags na solicitação devem estar nessa lista. No entanto, a solicitação pode incluir apenas uma das tags na lista. Por exemplo, você pode criar um usuário do IAM apenas com a etiqueta `Department=QualityAssurance`. No entanto, você não pode criar um usuário do IAM com a etiqueta `JobFunction=Employee` e a etiqueta `Project=core`. Para obter mais informações sobre o uso de `ForAllValues`, consulte Operadores de conjunto para chaves de contexto de vários valores.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM: gerar e recuperar relatórios de credenciais do IAM

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários gerem e baixem um relatório que liste todos os usuários do IAM na sua Conta da AWS. O relatório inclui o status das credenciais dos usuários, incluindo senhas, chaves de acesso, dispositivos MFA e certificados de assinatura. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática.

Para obter mais informações sobre relatórios de credenciais do , consulte Gerar relatórios de credenciais para sua Conta da AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:GetCredentialReport"
        ],
        "Resource": "*"
    }
}
```

IAM: permite gerenciar a associação de um grupo de forma programática e no console

Este exemplo mostra como é possível criar uma política baseada em identidade que permita atualizar a associação do grupo chamado `MarketingTeam`. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

O que essa política faz?

- A instrução do `ViewGroups` permite que o usuário liste todos os usuários e grupos no Console de gerenciamento da AWS. Ela também permite que o usuário visualize informações básicas sobre usuários da conta. Essas permissões devem estar nas suas respectivas instruções, pois não oferecem suporte ou não precisam especificar um ARN de recurso. Em vez disso, as permissões especificam `"Resource" : "*"`.
- A instrução `ViewEditThisGroup` permite que o usuário visualize informações sobre o grupo `MarketingTeam` e adicione e remova usuários do grupo.

Essa política não permite que o usuário visualize ou edite as permissões dos usuários ou do grupo `MarketingTeam`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM: gerenciar uma etiqueta específica

Este exemplo mostra como você pode criar uma política baseada em identidade que permita adicionar e remover a tag do IAM com a chave de tag `Department` de entidades do IAM (usuários e perfis). Esta política não limita o valor da etiqueta `Department`. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM: Passar uma função do IAM para um produto da AWS específico

Este exemplo mostra como você pode criar uma política baseada em identidade que permita passar qualquer perfil de serviço do IAM para o serviço Amazon CloudWatch. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Um perfil de serviço é um perfil do IAM que especifica um produto da AWS como a entidade principal que pode assumir o perfil. Isso permite que o serviço assuma a função e acesse recursos em outros serviços em seu nome. Para permitir que o Amazon CloudWatch assuma o perfil passado por você, é necessário especificar a entidade principal de serviço `cloudwatch.amazonaws.com` como a entidade principal na política de confiança do seu perfil. O escopo principal do serviço é definido pelo serviço. Para conhecer o escopo principal do serviço, consulte a documentação do serviço. Para alguns serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço. Pesquise `amazonaws.com` para visualizar a entidade principal do serviço.

Para saber mais sobre como passar uma função de serviço para um serviço, consulte Conceda permissões a um usuário para passar um perfil para um serviço da AWS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService":
 "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM: permite acesso somente leitura ao console do IAM sem relatórios

Este exemplo mostra como criar uma política baseada em identidade que permita que usuários do IAM realizem qualquer ação do IAM que comece com a string `Get` ou `List`. À medida que os usuários trabalham com o console, o console faz solicitações ao IAM para listar grupos, usuários, funções e políticas e para gerar relatórios sobre esses recursos.

O asterisco atua como um curinga. Quando você usa `iam:Get*` em uma política, as permissões resultantes incluem todas as ações do IAM que começam com `Get`, como `GetUser` e `GetRole`. Os caracteres curinga serão úteis se novos tipos de entidades forem adicionadas ao IAM no futuro. Nesse caso, as permissões concedidas pela política permitem automaticamente que o usuário liste e obtenha os detalhes sobre essas novas entidades.

Esta política não pode ser usada para gerar relatórios ou detalhes do último acesso do serviço. Para obter outra política que permita isso, consulte IAM: permite acesso somente leitura ao console do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

IAM: permite acesso somente leitura ao console do IAM

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que usuários do IAM realizem qualquer ação do IAM que comece com a string `Get`, `List` ou `Generate`. À medida que os usuários trabalham com o console do IAM, o console faz solicitações para listar grupos, usuários, funções e políticas e para gerar relatórios sobre esses recursos.

O asterisco atua como um curinga. Quando você usa `iam:Get*` em uma política, as permissões resultantes incluem todas as ações do IAM que começam com `Get`, como `GetUser` e `GetRole`. O uso de um caractere curinga será útil, principalmente se novos tipos de entidades forem adicionadas ao IAM no futuro. Nesse caso, as permissões concedidas pela política permitem automaticamente que o usuário liste e obtenha os detalhes sobre essas novas entidades.

Use esta política para acesso ao console que inclui permissões para gerar relatórios ou detalhes do último acesso do serviço. Para uma política distinta que não permite gerar ações, consulte IAM: permite acesso somente leitura ao console do IAM sem relatórios.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
```

```
            "iam:List*",
            "iam:Generate*"
        ],
        "Resource": "*"
    }
}
```

IAM: permite que os usuários do IAM gerenciem um grupo de forma programática e no console

Este exemplo mostra como é possível criar uma política baseada em identidade que permita que usuários do IAM específicos gerenciem o grupo `AllUsers`. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

O que essa política faz?

- A instrução `AllowAllUsersToListAllGroups` permite listar todos os grupos. Isso é necessário para a concessão de acesso ao console. Essa permissão deve estar em sua própria instrução, pois ela não oferece suporte a um ARN de recurso. Em vez disso, as permissões especificam `"Resource" : "*"`.
- A instrução `AllowAllUsersToViewAndManageThisGroup` permite todas as ações de grupo que podem ser executadas no tipo de recurso de grupo. Ela não permite a ação `ListGroupsForUser`, que pode ser executada em um tipo de recurso de usuário e não em um tipo de recurso de grupo. Para obter mais informações sobre os tipos de recursos que você pode especificar para uma ação do IAM, consulte Ações, recursos e chaves de condição do AWS Identity and Access Management.
- A declaração `LimitGroupManagementAccessToSpecificUsers` nega aos usuários com os nomes especificados o acesso para gravação e ações de grupo de gerenciamento de permissões. Quando um usuário especificado na política tentar fazer alterações no grupo, essa instrução não negará a solicitação. Essa solicitação é permitida pela instrução `AllowAllUsersToViewAndManageThisGroup`. Se outros usuários tentarem executar essas operações, a solicitação será negada. Você pode visualizar as ações do IAM definidas com os níveis de acesso Write (Gravação) ou Permissions management (Gerenciamento de permissões) ao criar essa política no console do IAM. Para fazer isso, alterne da guia JSON para a guia Visual

editor (Editor visual). Para obter mais informações sobre níveis de acesso, consulte Ações, recursos e chaves de condição do AWS Identity and Access Management.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
```

```
                }
            }
        }
    ]
}
```

IAM: permite configurar os requisitos de senha da conta de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que um usuário visualize e atualize os requisitos de senha da conta. Os requisitos de senha especificam os requisitos de complexidade e os períodos de rotação obrigatórios das senhas dos membros da conta. Esta política define permissões para acesso programático e do console.

Para saber como definir os requisitos de senha de sua conta, consulte Definir uma política de senhas de contas para usuários do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:UpdateAccountPasswordPolicy"
        ],
        "Resource": "*"
    }
}
```

IAM: acessar a API do simulador de política com base no caminho do usuário

Este exemplo mostra como você pode criar uma política baseada em identidade que permita usar a API do simulador de políticas apenas para usuários que têm o caminho `Department/Development`. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço`*

reservado em itálico na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> Para criar uma política que permite o uso do console do simulador de políticas para usuários que têm o caminho `Department/Development`, consulte IAM: acessar o console do simulador de políticas com base no caminho do usuário.

IAM: acessar o console do simulador de políticas com base no caminho do usuário

Este exemplo mostra como você pode criar uma política baseada em identidade que permita usar o console do simulador de política apenas para os usuários que têm o caminho `Department/Development`. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

IAM: permite que usuários do IAM autogerenciem um dispositivo com MFA

Este exemplo mostra como é possível criar uma política baseada em identidade que permita que os usuários do IAM autogerenciem seus dispositivos com autenticação multifator (MFA). Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática.

> **Note**
>
> Se um usuário do IAM com essa política não for autenticado por MFA, essa política negará o acesso a todas as ações da AWS, exceto aquelas necessárias para autenticar usando MFA. Se você adicionar essas permissões para um usuário que está conectado à AWS, pode ser necessário sair e fazer login novamente para visualizar essas alterações.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
```

```
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

IAM: permite que usuários do IAM alternem suas próprias credenciais de forma programática e no console

Este exemplo mostra como é possível criar uma política baseada em identidade que permita aos usuários do IAM atualizar suas próprias chaves de acesso, certificados de assinatura, credenciais específicas de serviço e senhas. Esta política define permissões para acesso programático e do console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Para saber como um usuário pode alterar sua própria senha no console, consulte the section called "Como um usuário do IAM altera a própria senha".

IAM: visualizar serviço acessado por último para uma política do AWS Organizations

Este exemplo mostra como você pode criar uma política baseada em identidade que permita visualizar as informações do último acesso do serviço para uma política específica do AWS Organizations. Essa política permite a recuperação dos dados para a política de controle de serviço (SCP - service control policy) com o ID `p-policy123`. A pessoa que gera e visualiza o relatório deve ser autenticada usando as credenciais da conta de gerenciamento do AWS Organizations. Essa política permite que o solicitante recupere os dados de qualquer entidade do AWS Organizations em sua organização. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Para obter informações importantes sobre as informações acessadas por último, incluindo as permissões necessárias, solução de problemas e regiões compatíveis, consulte Refinar permissões na AWS usando informações do último acesso.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
            }
        }
    ]
}
```

IAM: limita as políticas gerenciadas que podem ser aplicadas a um usuário, grupo ou função do IAM

Este exemplo mostra como você pode criar uma política do IAM que limita as políticas gerenciadas pelo cliente e gerenciadas pela AWS que podem ser aplicadas a um usuário, grupo ou perfil do IAM. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS: negar acesso a recursos fora da sua conta, exceto políticas do IAM gerenciadas pela AWS

O uso de `aws:ResourceAccount` em políticas baseadas em identidade podem afetar o usuário ou a capacidade da função de utilizar alguns serviços que exigem interação com recursos em contas pertencentes a um serviço.

Você pode criar uma política com uma exceção para contemplar as políticas do IAM gerenciadas pela AWS. Uma conta gerenciada por serviço fora do AWS Organizations é a proprietária das políticas gerenciadas pelo IAM. Há quatro ações do IAM que listam e recuperam políticas gerenciadas pela AWS. Use essas ações no elemento `NotAction` da instrução `AllowAccessToResourcesInSpecificAccountsAndSpecificService1` na política.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda: Permite que uma função Lambda acesse uma tabela do Amazon DynamoDB

Este exemplo mostra como você pode criar uma política baseada em identidade que permita o acesso de leitura e gravação a uma tabela específica do Amazon DynamoDB. A política também permite gravar arquivos de log no CloudWatch Logs. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Para usar esta política, anexe a política a uma função de serviço do Lambda. Uma função de serviço é uma função que você cria em sua conta para permitir que um serviço execute ações em seu nome. Essa função de serviço deve incluir o AWS Lambda como a entidade principal na política de confiança. Para obter detalhes sobre como usar essa política, consulte How to Create an AWS IAM Policy to Grant AWS Lambda Access to an Amazon DynamoDB Table no AWS Security Blog.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: permite acesso total ao banco de dados RDS em uma região específica

Este exemplo mostra como você pode criar uma política baseada em identidade que permita acesso total ao banco de dados do RDS em uma região específica. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:us-east-1:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS: permite a restauração de bancos de dados do RDS de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita restaurar bancos de dados do RDS. Esta política define permissões para acesso programático e do console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: permite aos proprietários de etiquetas acesso total aos recursos do RDS que eles etiquetaram

Este exemplo mostra como você pode criar uma política baseada em identidade que permita aos proprietários de tags total acesso aos recursos do RDS que eles marcaram. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
```

```
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner":
"${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
```

```
}
```

Amazon S3: permite que usuários do Amazon Cognito acessem objetos em seus buckets

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que os usuários do Amazon Cognito acessem objetos em um bucket específico do Amazon S3. Esta política permite acesso apenas a objetos com um nome que inclua `cognito`, o nome da aplicação e o ID da entidade principal federada representado pela variável ${cognito-identity.amazonaws.com:sub}. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

> **Note**
>
> O valor "sub" usado na chave de objeto não é o subvalor do usuário no grupo de usuários, é o ID de identidade associado ao usuário no grupo de identidades.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
```

```
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

O Amazon Cognito fornece autenticação, autorização e gerenciamento de usuários para suas aplicações Web e móveis. Seus usuários podem fazer login diretamente com um nome de usuário e senha ou por meio de terceiros, como Facebook, Amazon ou Google.

Os dois componentes principais do Amazon Cognito são os grupos de usuários e os grupos de identidades. Os grupos de usuários são diretórios de usuários que fornecem opções de cadastro e login para os usuários do seu aplicativo. Os grupos de identidade permitem que você conceda aos usuários acesso a outros serviços da AWS. Você pode usar grupos de identidades e grupos de usuários separadamente ou em conjunto.

Para obter mais informações sobre o Amazon Cognito, consulte o Guia do usuário do Amazon Cognito.

Amazon S3: permite que usuários federados acessem seus respectivos diretórios iniciais do Amazon S3 de forma programática e no console

Este exemplo mostra como é possível criar uma política baseada em identidade que permita que entidades principais federadas acessem seu próprio objeto de bucket do diretório inicial no S3. O diretório inicial é um bucket que inclui uma pasta home e pastas para entidades principais federadas individuais. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A variável `${aws:userid}` nessa política resulta em `role-id:specified-name`. A parte `role-id` do ID da entidade principal federada é um identificador exclusivo atribuído ao perfil do usuário federado durante a criação. Para obter mais informações, consulte Identificadores exclusivos. O `specified-name` é o parâmetro RoleSessionName passado para a solicitação `AssumeRoleWithWebIdentity` quando a entidade principal federada assumiu seu perfil.

Você pode visualizar o ID da função usando o comando da AWS CLI `aws iam get-role --role-name` *`specified-name`*. Por exemplo, imagine que você especifique o nome amigável `John`, e a CLI retorne o ID da função `AROAXXT2NJT7D3SIQN7Z6`. Nesse caso, o ID da entidade principal federada é `AROAXXT2NJT7D3SIQN7Z6:John`. Esta política permite que a entidade principal federada John acesse o bucket do Amazon S3 com o prefixo `AROAXXT2NJT7D3SIQN7Z6:John`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
```

```
                    "home/",
                    "home/${aws:userid}/*"
                ]
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}",
            "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:userid}/*"
        ]
    }
  ]
}
```

Amazon S3: acesso ao bucket do S3, mas bucket de produção negado sem MFA recente

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que um administrador do Amazon S3 acesse qualquer bucket, inclusive para atualização, adição e exclusão de objetos. No entanto, ela negará explicitamente o acesso ao bucket `amzn-s3-demo-bucket-production` se o usuário não tiver feito login usando Multi-Factor Authentication (MFA) nos últimos trinta minutos. Esta política concede as permissões necessárias para executar essa ação no console ou de forma programática usando a AWS CLI ou a API da AWS. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Esta política nunca permite o acesso programático ao bucket `amzn-s3-demo-bucket` usando chaves de acesso de usuário de longo prazo. Isso é feito usando a chave de condição `aws:MultiFactorAuthAge` com o operador de condição `NumericGreaterThanIfExists`. Essa condição de política retornará `true` se a MFA não estiver presente ou se a idade da MFA for maior do que 30 minutos. Nessas situações, o acesso será negado. Para acessar o bucket `amzn-s3-demo-bucket-production` de forma programática, o administrador do S3 deve usar credenciais temporárias geradas nos últimos 30 minutos usando a operação de API GetSessionToken.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-production/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
```

```
        }
    ]
}
```

Amazon S3: permite que usuários do IAM acessem seus diretórios base do S3 de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que usuários do IAM acessem seu próprio objeto de bucket do diretório inicial no S3. O diretório inicial é um bucket que inclui uma pasta `home` e pastas para usuários individuais. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Esta política não funcionará ao usar funções do IAM porque a variável `aws:username` não está disponível durante o uso das funções do IAM. Para obter detalhes sobre os principais valores-chave, consulte Valores de chave de principal.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
```

```
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
                "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3: limitar o gerenciamento a um bucket específico do S3

Este exemplo mostra como você pode criar uma política baseada em identidade que restrinja o gerenciamento de um bucket do Amazon S3 a esse bucket específico. Essa política concede permissão para realizar todas as ações do Amazon S3, mas nega acesso a todos os AWS service (Serviço da AWS), exceto o Amazon S3. Veja o exemplo a seguir. De acordo com essa política, você só pode acessar as ações do Amazon S3 que podem ser realizadas em um bucket do S3 ou em um recurso de objeto do S3. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

Se essa política for usada em combinação com outras políticas (como as políticas gerenciadas pela AWS AmazonS3FullAccess ou AmazonEC2FullAccess) que permitem ações negadas por esta política, o acesso será negado. Isso ocorre porque uma instrução de negação explícita tem precedência sobre instruções para permitir. Para obter mais informações, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso.

> **Warning**
>
> `NotAction` e `NotResource` são elementos de política avançados que devem ser usados com cuidado. Esta política nega o acesso a todos os produtos da AWS, exceto o Amazon S3. Se você anexar essa política a um usuário, quaisquer outras políticas que concedam permissões para outros serviços são ignoradas e o acesso é negado.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

Conceda acesso de leitura e gravação aos objetos do bucket do Amazon S3

Este exemplo mostra como você pode criar uma política baseada em identidade que permita o acesso de `Read` e `Write` a objetos em um bucket específico do Amazon S3. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática. Para usar esta política, substitua o *texto do espaço reservado em itálico*

na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A ação `s3:*Object` usa um curinga como parte do nome da ação. A instrução `AllObjectActions` permite `GetObject`, `DeleteObject`, `PutObject` e qualquer outra ação do Amazon S3 que termine com a palavra “Object” (Objeto).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> Para conceder acesso de `Read` e `Write` a um objeto em um bucket do Amazon S3 e também incluir permissões adicionais para acesso ao console, consulte Amazon S3: permite acesso de leitura e gravação a objetos em um bucket do S3 de forma programática e no console.

Amazon S3: permite acesso de leitura e gravação a objetos em um bucket do S3 de forma programática e no console

Este exemplo mostra como você pode criar uma política baseada em identidade que permita o acesso de `Read` e `Write` a objetos em um bucket específico do S3. Esta política define permissões para acesso programático e do console. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

A ação `s3:*Object` usa um curinga como parte do nome da ação. A instrução `AllObjectActions` permite `GetObject`, `DeleteObject`, `PutObject` e qualquer outra ação do Amazon S3 que termine com a palavra “Object” (Objeto).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
```

```
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
        }
    ]
}
```

Gerenciar políticas do IAM

O IAM fornece as ferramentas para criação e gerenciamento de todos os tipos de políticas do IAM (políticas gerenciadas e em linha). Para adicionar permissões a uma identidade do IAM (usuário, grupo ou função do IAM), crie uma política,valide a política, e, em seguida, anexe a política à identidade. Você pode anexar várias políticas a uma identidade e cada política pode conter várias permissões.

Tópicos

- Recursos adicionais
- Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente
- Validação de política do IAM
- Testar políticas do IAM com o simulador de políticas do IAM
- Adicionar e remover permissões de identidade do IAM
- Versionamento de políticas do IAM
- Editar políticas do IAM
- Excluir políticas do IAM
- Refinar permissões na AWS usando informações do último acesso

Recursos adicionais

Os recursos a seguir podem ajudar você a saber mais sobre as políticas da AWS.

- Para obter mais informações sobre os diferentes tipos de políticas do IAM, consulte Políticas e permissões no AWS Identity and Access Management.
- Para obter informações gerais sobre o uso de políticas no IAM, consulte Gerenciamento de acesso para recursos da AWS.

- Para obter informações sobre como usar o IAM Access Analyzer para gerar uma política do IAM baseada na atividade de acesso para uma entidade, consulte Geração de políticas do IAM Access Analyzer.
- Para obter informações sobre como as permissões são avaliadas quando várias políticas estão em vigor para determinada identidade do IAM, consulte Lógica da avaliação de política.
- O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para ter mais informações, consulte IAM e cotas do AWS STS.

Defina permissões personalizadas do IAM com políticas gerenciadas pelo cliente

As políticas definem permissões para identidades ou recursos na AWS. Você pode usar o Console de gerenciamento da AWS, a AWS CLI, ou a API da AWS para criar políticas gerenciadas pelo cliente no IAM. As políticas gerenciadas pelo cliente são políticas autônomas que você administra na sua própria Conta da AWS. Você pode anexar as políticas a identidades (usuários, grupos e perfis) na sua conta Conta da AWS.

Uma política anexada a uma identidade no IAM é conhecida como uma política baseada em identidade. As políticas baseadas em identidade podem incluir políticas gerenciadas pela AWS, políticas gerenciadas pelo cliente e políticas em linha. As políticas gerenciadas pela AWS são criadas e gerenciadas pela AWS. Você pode usá-las, mas não pode gerenciá-las. Uma política em linha é aquela que você cria e incorpora diretamente em um grupo de usuários, usuário ou perfil do IAM. As políticas em linha não podem ser reutilizadas em outras identidades ou gerenciadas fora da identidade onde existem. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

Em geral, é melhor usar políticas gerenciadas pelo cliente em vez de políticas em linha ou políticas gerenciadas pela AWS. As políticas gerenciadas pela AWS costumam fornecer amplas permissões administrativas ou somente leitura. Para maior segurança, conceda privilégio mínimo. Ele concede apenas as permissões necessárias para executar tarefas de trabalho específicas.

Quando você cria ou edita políticas do IAM, a AWS pode executar automaticamente a validação de políticas para ajudar você a criar uma política eficaz com o mínimo privilégio em mente. No Console de gerenciamento da AWS, o IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de política do

IAM. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer.

Você pode usar o Console de gerenciamento da AWS, a AWS CLI ou a API da AWS para criar políticas gerenciadas pelo cliente no IAM. Para obter mais informações sobre o uso de modelos do CloudFormation para adicionar ou atualizar políticas, consulte a Referência a tipos de recursos do AWS Identity and Access Management, no Guia do usuário do CloudFormation.

Tópicos

- Criar políticas do IAM (console)
- Criar políticas do IAM (AWS CLI)
- Criar políticas do IAM (API da AWS)

Criar políticas do IAM (console)

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. Você pode usar a Console de gerenciamento da AWS para criar políticas gerenciadas pelo cliente no IAM. Políticas gerenciadas pelo cliente são políticas autônomas que você administra na sua própria Conta da AWS. Você pode anexar as políticas a identidades (usuários, grupos e perfis) na sua conta Conta da AWS.

O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Tópicos

- Criação de políticas do IAM
- Criar políticas usando o editor de JSON
- Criar políticas com o editor visual
- Importar políticas gerenciadas existentes

Criação de políticas do IAM

Você pode criar uma política gerenciada pelo cliente no Console de gerenciamento da AWS usando um dos seguintes métodos:

- JSON: cole e personalize uma política baseada em identidade de exemplo publicada.

- Editor visual: crie uma nova política do zero no editor visual. Se você usar o editor visual, não precisará entender a sintaxe JSON.
- Importar: importe e personalize uma política gerenciada na sua conta. Você pode importar uma política gerenciada da AWS ou uma política gerenciada pelo cliente criada anteriormente.

O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Criar políticas usando o editor de JSON

Você pode digitar ou colar políticas em JSON escolhendo a opção JSON. Este método é útil para copiar um exemplo de política para usar na sua conta. Você também pode digitar o seu próprio documento de política JSON no editor JSON. Você também pode usar a opção JSON para alternar entre o editor visual e JSON para comparar as visualizações.

Quando você cria ou edita uma política no editor JSON, o IAM executa a validação da política para ajudar você a criar uma política eficaz. O IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de política adicionais com recomendações práticas para ajudar você a refinar ainda mais a política.

Uma política JSON consiste em uma ou mais instruções. Cada instrução deve conter todas as ações que compartilham o mesmo efeito (`Allow` ou `Deny`) e oferecem suporte aos mesmos recursos e condições. Se uma ação exigir que você especifique todos os recursos (`"*"`) e outra ação oferecer suporte ao nome de recurso da Amazon (ARN) de um recurso específico, elas devem ficar em duas instruções JSON separadas. Para obter detalhes sobre formatos de ARN, consulte Nome de recurso da Amazon (ARN) no Guia de Referência geral da AWS. Para obter informações gerais sobre políticas do IAM, consulte Políticas e permissões no AWS Identity and Access Management. Para obter informações sobre a linguagem de políticas do IAM, consulte Referência de política JSON do IAM.

Para usar o editor de políticas JSON para criar uma política

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON.

5. Digite ou cole um documento de política JSON. Para obter detalhes sobre a linguagem de políticas do IAM, consulte Referência de política JSON do IAM.
6. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a validação de política e depois escolha Próximo.

> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

7. (Opcional) Ao criar ou editar uma política no Console de gerenciamento da AWS, você pode gerar um modelo de política JSON ou YAML que pode ser usado em modelos do CloudFormation.

   Para isso, no editor de políticas, escolha Ações e depois escolha Gerar modelo do CloudFormation. Para saber mais sobre o CloudFormation, consulte Referência de tipos de recurso do AWS Identity and Access Management no Guia do usuário do AWS CloudFormation.
8. Quando terminar de adicionar as permissões à política, escolha Avançar.
9. Na página Revisar e criar, digite um nome de política e uma descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política.
10. (Opcional) Adicione metadados à política associando tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
11. Escolha Criar política para salvar sua nova política.

Depois de criar uma política, você pode anexá-la aos usuários, grupos ou funções. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

Criar políticas com o editor visual

O editor visual no console do IAM oferece orientação para a criação de uma política sem que seja necessário escrever usando a sintaxe JSON. Para visualizar um exemplo de como usar o editor visual para criar uma política, consulte the section called “Controle de acesso a identidades”.

Para usar o editor visual para criar uma política

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Criar política.
4. Na seção Editor de políticas, localize a seção Selecionar um serviço e escolha um serviço da AWS. Você pode usar a caixa de pesquisa na parte superior para limitar os resultados da lista de serviços. Você pode escolher apenas um serviço em um bloco de permissões no editor visual. Para conceder acesso a mais de um serviço, adicione vários blocos de permissões escolhendo Adicionar mais permissões.
5. Em Ações permitidas, escolha as ações a serem adicionadas à política. Você pode escolher as ações das seguintes maneiras:

   - Marque a caixa de seleção para todas as ações.
   - Escolha adicionar ações para digitar o nome de uma ação específica. Você pode usar curingas (*) para especificar várias ações.
   - Selecione um dos grupos de Nível de acesso para escolher todas as ações do nível de acesso (por exemplo, Leitura, Gravação ou Lista).
   - Expanda cada um dos grupos de Access level para escolher as ações individuais.

   Por padrão, a política que você está criando permite as ações que você escolhe. Para negar as ações escolhidas, selecione Alternar para negar permissões. Como o IAM nega por padrão, a prática recomendada de segurança é que você conceda permissões somente para as ações e os recursos de que um usuário precisa. Você só deverá criar uma instrução JSON para negar permissões se desejar substituir, separadamente, uma permissão que é concedida por uma outra instrução ou política. Recomendamos que você limite ao mínimo o número de permissões de negação, pois elas podem aumentar a dificuldade de solucionar problemas nas permissões.
6. Para Recursos, se o serviço e as ações que você selecionou nas etapas anteriores não oferecerem suporte à escolha de recursos específicos, todos os recursos serão permitidos e você não poderá editar esta seção.

   Se você escolher uma ou mais ações que ofereçam suporte a permissões no nível de recursos, o editor visual listará esses recursos. Você poderá expandir Recursos para especificar os recursos para sua política.

É possível especificar recursos das seguintes maneiras:

- Selecione Adicionar ARNs para especificar os recursos pelos nomes dos recursos da Amazon (ARN). Você pode usar o editor de ARN visual ou listar os ARNs manualmente. Para obter mais informações sobre a sintaxe do ARN, consulte Nome de recurso da Amazon (ARN) no Guia de Referência geral da AWS. Para obter informações sobre como usar ARNs no elemento `Resource` de uma política, consulte Elementos de política JSON do IAM: Resource.
- Escolha Qualquer um nesta conta ao lado de um recurso para conceder permissões a qualquer recurso desse tipo.
- Escolha Todos os recursos para escolher todos os recursos para o serviço.

7. (Opcional) Escolha Condições de solicitação - opcional para adicionar condições à política que você está criando. As condições limitam o efeito de uma instrução de política JSON. Por exemplo, você pode especificar que um usuário só tem permissão para executar ações nos recursos quando sua solicitação ocorrer em um determinado período. Você também pode usar as condições mais usadas para limitar se um usuário deve ser autenticado usando um dispositivo Multi-Factor Authentication (MFA – Autenticação multifator). Ou você pode exigir que a solicitação tenha origem em um determinado intervalo de endereços IP. Para obter uma lista completa das chaves de contexto que podem ser usadas na condição de uma política, consulte Ações, recursos e chaves de condição para serviços da AWS na Referência de autorização de serviços.

   Você pode escolher as condições das seguintes maneiras:

   - Use as caixas de seleção para selecionar condições comumente utilizadas.
   - Escolha Adicionar outra condição para especificar outras condições. Escolha a Condition Key, o Qualifier e o Operator da condição e, em seguida, digite um Value. Para adicionar mais de um valor, escolha Adicionar. Você pode considerar os valores como sendo conectados por um operador lógico "OR". Quando terminar, selecione Adicionar condição.

   Para adicionar mais de uma condição, escolha novamente Adicionar outra condição. Repita conforme necessário. Cada condição se aplica apenas a um bloco de permissões do editor visual. Todas as condições devem ser verdadeiras para que o bloco de permissões seja considerado uma correspondência. Em outras palavras, considere as condições como sendo conectadas por um operador lógico "AND".

Para obter mais informações sobre o elemento Condição, consulte Elementos de política JSON do IAM: Condition no Referência de política JSON do IAM.

8. Para adicionar mais blocos de permissão, escolha Adicionar mais permissões. Para cada bloco, repita as etapas de 2 a 5.

> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

9. (Opcional) Ao criar ou editar uma política no Console de gerenciamento da AWS, você pode gerar um modelo de política JSON ou YAML que pode ser usado em modelos do CloudFormation.

   Para isso, no editor de políticas, escolha Ações e depois escolha Gerar modelo do CloudFormation. Para saber mais sobre o CloudFormation, consulte Referência de tipos de recurso do AWS Identity and Access Management no Guia do usuário do AWS CloudFormation.
10. Quando terminar de adicionar as permissões à política, escolha Avançar.
11. Na página Revisar e criar, digite um nome de política e uma descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ter certeza de que você concedeu as permissões que pretendia.
12. (Opcional) Adicione metadados à política associando tags como pares de chave-valor. Para obter mais informações sobre como usar tags no IAM, consulte Tags para recursos do AWS Identity and Access Management.
13. Escolha Criar política para salvar sua nova política.

Depois de criar uma política, você pode anexá-la aos usuários, grupos ou funções. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

Importar políticas gerenciadas existentes

Uma maneira fácil de criar uma nova política é importar uma política gerenciada existente para sua conta que tenha pelo menos algumas das permissões de que você precisa. Em seguida, você pode personalizar a política de acordo seus novos requisitos.

Não é possível importar uma política em linha. Para saber mais sobre a diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.

Para importar uma política gerenciada existente no editor visual

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Criar política.
4. Em Editor de políticas, escolha Visual e depois, no lado direito da página, escolha Ações e depois Importar política.
5. Na janela Importar política, escolha as políticas gerenciadas que mais se aproximam da política que você deseja incluir na nova política. Você pode usar a caixa de pesquisa na parte superior para limitar os resultados da lista de políticas.
6. Escolha Importar política.

   As políticas importadas são adicionadas em novos blocos de permissões na parte inferior da política.
7. Use o Editor visual ou escolha JSON para personalizar a política. Escolha Próximo.

   > **Note**
   >
   > Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.
8. Na página Revisar e criar, digite um nome de política e uma descrição (opcional) para a política que você está criando. Você não poderá editar essas configurações mais tarde. Revise Permissões definidas nessa política e depois escolha Criar política para salvar seu trabalho.

Para importar uma política gerenciada existente no editor JSON

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.

3. Escolha Criar política.
4. Na seção Editor de políticas, escolha a opção JSON e depois, no lado direito da página, escolha Ações e depois Importar política.
5. Na janela Importar política, escolha as políticas gerenciadas que mais se aproximam da política que você deseja incluir na nova política. Você pode usar a caixa de pesquisa na parte superior para limitar os resultados da lista de políticas.
6. Escolha Importar política.

   As instruções das políticas importadas são adicionadas na parte inferior da sua política JSON.
7. Personalize a política em JSON. Resolva os avisos de segurança, as mensagens erros ou os avisos gerais gerados durante a validação de política, e depois escolha Próximo. Ou personalize sua política no Visual editor (Editar visual). Escolha Próximo.

   > **Note**
   >
   > Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.
8. Na página Revisar e criar, digite um nome de política e uma descrição (opcional) para a política que você está criando. Você não poderá editá-los mais tarde. Revise Permissões definidas nessa política e escolha Criar política para salvar seu trabalho.

Depois de criar uma política, você pode anexá-la aos usuários, grupos ou funções. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.

Criar políticas do IAM (AWS CLI)

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. Você pode usar a AWS CLI para criar políticas gerenciadas pelo cliente no IAM. Políticas gerenciadas pelo cliente são políticas autônomas que você administra na sua própria Conta da AWS. Como prática recomendada, recomendamos que você use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais. Ao validar suas políticas, você pode resolver quaisquer erros ou recomendações antes de anexá-las às identidades (usuários, grupos e perfis) na sua Conta da AWS.

O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para ter mais informações, consulte IAM e cotas do AWS STS.

Criação de políticas do IAM (AWS CLI)

Você pode criar uma política do IAM gerenciada pelo cliente ou uma política em linha usando a AWS Command Line Interface (AWS CLI).

Para criar uma política gerenciada pelo cliente (AWS CLI)

Use o seguinte comando:

- create-policy

Como criar uma política em linha em uma identidade do IAM (usuário, grupo ou função) (AWS CLI)

Use um dos seguintes comandos:

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> Não é possível usar o IAM para incorporar uma política em linha para uma função vinculada ao serviço.

Para validar uma política gerenciada pelo cliente (AWS CLI)

Use o seguinte comando do IAM Access Analyzer:

- validate-policy

Criar políticas do IAM (API da AWS)

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. Você pode usar a API da AWS para criar políticas gerenciadas pelo cliente no IAM. Políticas gerenciadas pelo cliente são políticas autônomas que você administra na sua própria Conta

da AWS. Como prática recomendada, recomendamos que você use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais. Ao validar suas políticas, você pode resolver quaisquer erros ou recomendações antes de anexá-las às identidades (usuários, grupos e perfis) na sua Conta da AWS.

O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Como criar políticas do IAM (API da AWS)

Você pode criar uma política gerenciada pelo cliente do IAM ou uma política em linha usando a API da AWS.

Para criar uma política gerenciada pelo cliente (API da AWS)

Chame a seguinte operação:

- CreatePolicy

Para criar uma política em linha para uma identidade do IAM (usuário, grupo ou função) (API da AWS)

Chame uma das seguintes operações:

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> Não é possível usar o IAM para incorporar uma política em linha para uma função vinculada ao serviço.

Para validar uma política gerenciada pelo cliente (API da AWS)

Chame a seguinte operação do IAM Access Analyzer:

- ValidatePolicy

Validação de política do IAM

Uma política é um documento JSON que usa a gramática de política do IAM. Quando você anexa uma política a uma entidade do IAM, como um usuário, um grupo ou uma função, ela concede permissões a essa entidade.

Quando você cria ou edita políticas de controle de acesso do IAM usando o Console de gerenciamento da AWS, a AWS as analisa automaticamente para garantir que estejam em conformidade com a gramática de política do IAM. Se a AWS determinar que uma política não está em conformidade com a gramática, ela solicitará que você corrija a política.

O IAM Access Analyzer fornece verificações de política adicionais com recomendações para ajudar você a refinar ainda mais a política. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer. Para visualizar uma lista dos avisos, erros e sugestões retornados pelo IAM Access Analyzer, consulte Referência de verificação de política do Access Analyzer.

Escopo da validação

A AWS verifica a sintaxe e a gramática da política JSON. Ela também verifica se seus ARNs estão formatados corretamente e se os nomes de ação e as chaves de condição estão corretos.

Acesso à validação de política

As políticas são validadas automaticamente quando você cria uma política JSON ou edita uma política existente no Console de gerenciamento da AWS. Se a sintaxe da política não for válida, você receberá uma notificação e deverá corrigir o problema antes de continuar. As descobertas da validação da política do IAM Access Analyzer são retornadas automaticamente no Console de gerenciamento da AWS se você tiver permissões para `access-analyzer:ValidatePolicy`. Você também pode validar políticas usando a API da AWS ou a AWS CLI.

Políticas existentes

Você pode ter políticas existentes que não são válidas porque foram criadas ou salvas pela última vez antes das atualizações mais recentes do mecanismo de política. Como prática recomendada, recomendamos que você use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais. Recomendamos que você abra suas políticas existentes e analise os resultados da validação da política que são gerados. Você não pode editar e salvar as políticas existentes sem corrigir erros de sintaxe da política.

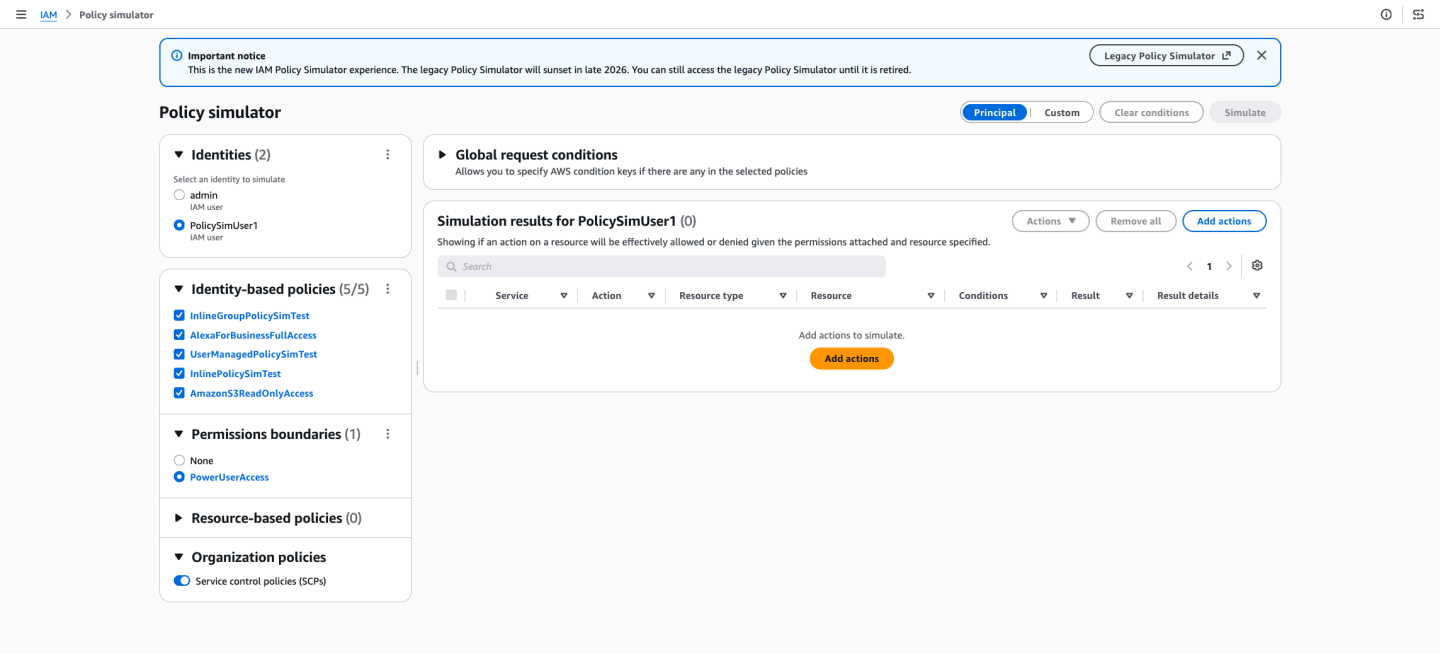
Testar políticas do IAM com o simulador de políticas do IAM

Com o simulador de políticas do IAM, você pode testar políticas baseadas em identidade, limites de permissões do IAM, políticas de controle de serviços (SCPs) e políticas baseadas em recurso que você fornecer. Você simula como a AWS avaliaria uma solicitação em relação a essas políticas, sem enviar uma solicitação real para nenhum serviço da AWS. Cada simulação retorna um resultado de autorização binária para cada ação e recurso que você testa: a ação é permitida ou negada. Quando o resultado é uma permissão ou uma negação explícita, o simulador também mostra qual política produziu esse resultado, para que você possa entender e corrigir a causa.

Para obter mais informações sobre como e por que usar políticas do IAM, consulte Políticas e permissões no AWS Identity and Access Management.

> **Important**
>
> Os resultados do simulador de políticas podem ser diferentes do seu ambiente da AWS ativo. Recomendamos comparar suas políticas em relação ao seu ambiente da AWS ativo depois de testar com o simulador de políticas para confirmar que obteve os resultados desejados. Para obter mais informações, consulte Como o simulador de políticas do IAM funciona.

A captura de tela a seguir mostra o simulador de políticas do IAM, em que você seleciona uma identidade e as respectivas políticas e escolhe as ações e os recursos a serem testados antes de executar a simulação.



Tópicos

- Modos do simulador de políticas
- O que é possível fazer com o simulador de políticas
- Como o simulador de políticas do IAM funciona
- Como funciona a avaliação da chave de condição no simulador de políticas do IAM
- Permissões para usar o simulador de políticas
- Como simular políticas

Modos do simulador de políticas

O simulador de políticas tem dois modos. Escolha o modo que corresponda ao fato de você querer testar políticas que já existem na sua conta ou políticas que ainda estão sendo elaboradas.

Modo Entidade principal

No modo Entidade principal, você testa as políticas que já estão anexadas a um usuário, perfil ou grupo do IAM existente. Você escolhe uma identidade para a simulação, e o simulador avalia as políticas associadas a essa identidade. Você também pode incluir ou excluir políticas de identidade personalizadas ou limites de permissões na simulação. Essas adições são simuladas como se estivessem vinculadas à entidade principal, sem realmente serem anexadas ao IAM; portanto, sua conta permanece inalterada.

Modo Personalizado

No modo Personalizado, você testa uma ou mais políticas que você escreve ou cola. Ele é útil para verificar uma política antes de anexá-la. As políticas personalizadas são simuladas isoladamente e não são salvas na sua conta.

Para obter um passo a passo detalhado sobre como selecionar um modo e executar uma simulação, consulte Como simular políticas.

O que é possível fazer com o simulador de políticas

Veja algumas ações comuns que você pode executar com o simulador de políticas. Um guia passo a passo mais detalhado e uma distinção entre os comportamentos da API e do console podem ser encontrados em Como simular políticas.

- Teste as políticas baseadas em identidade, se elas já foram anexadas a um usuário, grupo ou perfil do IAM, ou novas políticas que você escreve ou cola. Para uma identidade anexada com mais de uma política, você pode testar todas as políticas em conjunto ou selecionar políticas individuais e ver quais ações são permitidas ou negadas para recursos específicos. As novas políticas são avaliadas como se estivessem anexadas a uma identidade, sem realmente serem anexadas à sua conta; elas são usadas somente na simulação e não são salvas na sua conta.
- Teste o efeito dos controles restritivos no acesso. É possível simular um limite de permissões por vez. Se a sua Conta da AWS for membro de uma organização no AWS Organizations, você também poderá testar o impacto das políticas de controle de serviço (SCPs) em suas políticas baseadas em identidade. O simulador informa se um SCP permite ou nega uma ação, mas, por motivos de segurança, ele não mostra as instruções correspondentes de um SCP da mesma forma que faz com outros tipos de política.
- Teste o efeito de uma política baseada em recursos que você fornece. Exceto no console, o simulador não recupera a política de um recurso para você. Para incluir uma política baseada em recursos em uma simulação, você fornece a política junto com o recurso.
- Teste serviços, ações e recursos específicos em condições reais e identifique o que determina o resultado. Por exemplo, você pode testar se uma política permite as ações `ListAllMyBuckets`, `CreateBucket` e `DeleteBucket` no Amazon S3 em um bucket específico e fornecer chaves de contexto, como um endereço IP ou uma data, que são referenciados em elementos de `Condition` nas políticas que estão sendo testadas. Para cada resultado, o simulador identifica qual instrução específica permite ou nega acesso; instruções correspondentes podem vir de políticas baseadas em identidade e de políticas baseadas em recursos que você fornece.

Como o simulador de políticas do IAM funciona

Quando você executa uma simulação, o simulador de políticas avalia as políticas no escopo da solicitação, quaisquer SCPs no escopo e as entradas que você fornece, como ações, recursos e valores de chave de contexto. No modo Personalizado, você fornece as políticas diretamente. No modo Entidade principal, o simulador usa as políticas anexadas à identidade selecionada e você pode incluir ou excluir mais políticas da simulação. O resultado é binário para cada ação e recurso: permitido ou negado.

O simulador de políticas difere do ambiente de produção da AWS das seguintes formas:

- Como o simulador de políticas não faz uma solicitação de serviço da AWS real, você pode testar com segurança as solicitações que, do contrário, poderiam fazer alterações indesejadas em seu ambiente de produção da AWS. Ele não usa valores de chave de contexto reais da produção e,

como as ações não são realmente executadas, ele não retorna nenhuma resposta do serviço. O único resultado é se cada ação solicitada seria permitida ou negada.

- Se você editar uma política no simulador de políticas, a alteração afetará apenas a simulação. A política correspondente em sua Conta da AWS permanecerá inalterada.
- O simulador de políticas avalia SCPs, incluindo chaves de condição e escopo de recursos em instruções de negação, mas não oferece suporte a políticas de controle de recursos (RCPs). Os resultados da simulação ainda podem diferir do comportamento real para algumas configurações avançadas, como políticas de endpoint da VPC, encadeamento de perfis ou várias políticas baseadas em recursos em um único recurso.
- Simulações entre contas retornam decisões por tipo de política (política de identidade e política de recursos) em `EvalDecisionDetails`.

Como funciona a avaliação da chave de condição no simulador de políticas do IAM

Muitas políticas usam elementos de `Condition` que dependem de chaves de condição, como `aws:RequestedRegion` ou `aws:MultiFactorAuthPresent`. Quando você executa uma simulação, o simulador de políticas obtém valores para as chaves de condição de duas fontes: valores que o simulador preenche automaticamente e valores que você fornece como entrada de simulação. Se um valor não atender a uma condição, a ação será negada.

Contexto que o simulador preenche automaticamente

O simulador preenche as seguintes chaves de contexto de entidade principal e organizacional para a avaliação:

- Contexto da entidade principal: `aws:PrincipalAccount`, `aws:PrincipalId`, `aws:PrincipalType`, `aws:Type`, `aws:UserId`, `aws:UserName`
- Contexto da organização: `aws:PrincipalOrgID`, `aws:PrincipalOrgMasterAccountId`, `aws:PrincipalOrgPaths`

Como essas chaves são preenchidas para você, as SCPs e outras políticas que usam condições de entidade principal ou organizacionais são avaliadas corretamente. Você fornece os valores para todas as outras chaves de condição que suas políticas referenciam. Revise os elementos de `Condition` nas políticas que você está testando e forneça valores para todas as chaves que não estejam na lista acima.

Chaves de condição globais e suporte de serviço

Para políticas baseadas em identidade e políticas baseadas em recursos, o simulador não determina quais serviços oferecem suporte a uma determinada chave de condição global para autorização. Por exemplo, o simulador não identifica quando um serviço não oferece suporte a `aws:TagKeys`. Por outro lado, as políticas de controle de serviços (SCPs) são avaliadas com uma lógica ciente do serviço, incluindo chaves de condição globais como `aws:RequestedRegion` e `aws:PrincipalAccount`. Para obter mais informações sobre essas chaves de condição, consulte AWSChaves de contexto de condições globais da.

Chaves de condição em políticas de controle de serviços (SCPs)

O simulador avalia as chaves de condição e o escopo dos recursos em SCPs. A forma como você fornece valores para essas chaves depende de como você executa a simulação:

- Com a AWS CLI ou a API da AWS, você pode fornecer valores para chaves de condição que são referenciadas por SCPs.
- No console do IAM, você só pode definir valores para chaves de condição que aparecem em suas políticas baseadas em identidade, limites de permissões ou políticas baseadas em recurso. Se uma chave de condição for referenciada somente por uma SCP, você não poderá definir um valor para ela no console. No entanto, se a mesma chave de condição também aparecer em uma dessas outras políticas, o valor definido será usado em toda a avaliação, incluindo a SCP.

Por motivos de segurança, a avaliação da SCP não retorna valores de contexto ausentes. Você ainda recebe a decisão de permitir ou negar a solicitação.

Permissões para usar o simulador de políticas

As permissões necessárias dependem de você usar a AWS CLI ou a API da AWS, do console do IAM e do modo usado.

Console

Você precisa de permissão para chamar `iam:GetContextKeysForCustomPolicy` para usar o simulador de políticas do IAM no console.

As permissões adicionais necessárias dependem do modo. No modo Entidade principal, você precisa de permissões de leitura adicionais para listar as identidades em sua conta e ler as políticas anexadas. Para reduzir o que você precisa conceder, use o modo Personalizado se precisar testar apenas as políticas que você cola.

Modo Personalizado

Para usar o modo Personalizado, no qual você cola as políticas que deseja testar, você também precisa de `iam:SimulateCustomPolicy`

Modo Entidade principal

Para usar o modo Entidade principal, em que o console carrega e exibe as identidades e políticas em sua conta, você precisa da permissão comum e do seguinte. Esse é o conjunto maior, porque o console precisa descobrir identidades e ler as políticas anexadas.

Simular:

- `iam:SimulatePrincipalPolicy`

Listar identidades:

- `iam:ListUsers`, `iam:ListRoles`, `iam:ListGroups`
- `iam:ListGroupsForUser`

Listar políticas anexadas:

- `iam:ListPolicies`, `iam:ListUserPolicies`, `iam:ListRolePolicies`, `iam:ListGroupPolicies`
- `iam:ListAttachedUserPolicies`, `iam:ListAttachedRolePolicies`, `iam:ListAttachedGroupPolicies`

Obter documentos de política:

- `iam:GetPolicyVersion`, `iam:GetUserPolicy`, `iam:GetRolePolicy`, `iam:GetGroupPolicy`

Obter limites de permissões:

- `iam:GetUser`, `iam:GetRole`

Permissões adicionais para o modo Entidade principal, a fim de simular políticas baseadas em recursos

Para incluir a política de um recurso em uma simulação do modo Entidade principal, adicione a permissão de leitura para esse tipo de recurso:

- `s3:GetBucketPolicy`, `sns:GetTopicAttributes`, `sqs:GetQueueUrl`, `sqs:GetQueueAttributes`, `glacier:GetVaultAccessPolicy`

Para ver exemplos das políticas que concedem acesso ao simulador, consulte the section called "Exemplo de políticas do AWS Identity and Access Management (IAM)".

AWS CLI and AWS API

Para a AWS CLI e a API da AWS, você precisa apenas da permissão para chamar a ação de simulação para o tipo de política que deseja testar. Se você usar `iam:GetContextKeysForCustomPolicy` e `iam:GetContextKeysForPrincipalPolicy` para descobrir primeiro as chaves de contexto, conceda permissões também para essas ações. Por motivos de segurança, as ações são divididas nos dois modos a seguir. Conceda somente as permissões que correspondam ao que você deseja simular.

Para simular políticas que você passa diretamente para a API como strings (políticas personalizadas)

- `iam:SimulateCustomPolicy`
- `iam:GetContextKeysForCustomPolicy` (somente se você usar a etapa de descoberta da chave de contexto)

Essas ações permitem testar políticas que ainda não foram anexadas a um usuário, grupo de usuários ou perfil. Você passa as políticas como strings e elas são usadas somente na simulação.

Para simular políticas anexadas a um usuário, grupo de usuários, perfil ou recurso do IAM

- `iam:SimulatePrincipalPolicy`
- `iam:GetContextKeysForPrincipalPolicy` (somente se você usar a etapa de descoberta da chave de contexto)

Como essas ações podem revelar as permissões atribuídas a outras entidades do IAM, considere restringir quem pode chamá-las. Por exemplo, para fornecer a um usuário chamado Bob permissão

para simular uma política atribuída a uma usuária chamada Alice, conceda ao Bob acesso ao recurso `arn:aws:iam::777788889999:user/alice`.

Para visualizar um exemplo de política que permite usar a API do simulador de políticas para políticas anexadas e não anexadas na Conta da AWS atual, consulte IAM: acessar a API do simulador de política. Para ver um exemplo de política com escopo definido para usuários com um caminho específico, consulte IAM: acessar a API do simulador de política com base no caminho do usuário.

Como simular políticas

Você pode executar simulações no console do IAM ou com a AWS CLI e a API da AWS. Para obter uma visão geral do que o simulador avalia, dos dois modos e de como ele alcança um resultado, consulte Testar políticas do IAM com o simulador de políticas do IAM. Para as permissões necessárias, consulte Permissões para usar o simulador de políticas.

Tópicos

- Simular políticas (console)
- Simular políticas (AWS CLI e API da AWS)

Simular políticas (console)

Você pode executar simulações no simulador de políticas no console do IAM. No painel de navegação, escolha Simulador de políticas.

> **Note**
>
> O console autônomo do simulador de políticas em https://policysim.aws.amazon.com/ não é mais mantido. Use o simulador de políticas no console do IAM.

Modo Entidade principal

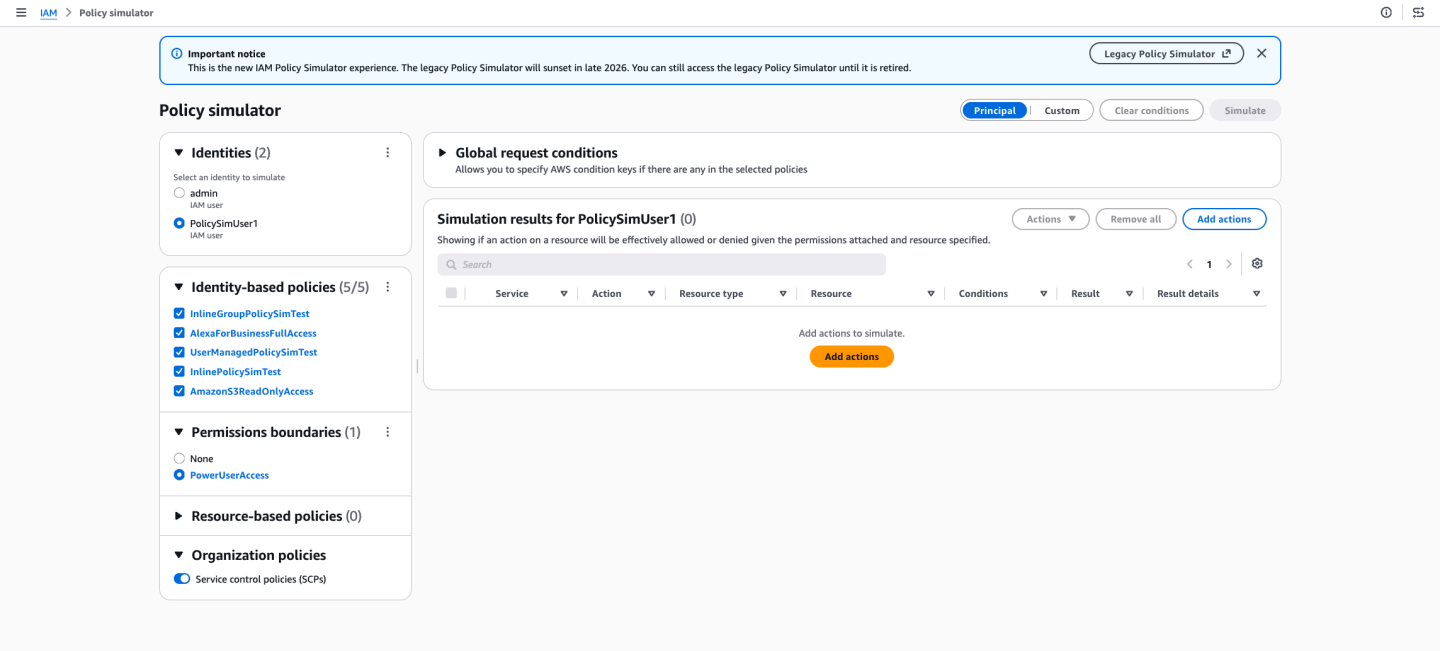
Selecionar uma identidade

Você começa selecionando uma identidade. Você pode selecionar usuários, perfis ou grupos do IAM nessa conta. O painel lateral preencherá as políticas anexadas a essas identidades após sua seleção.

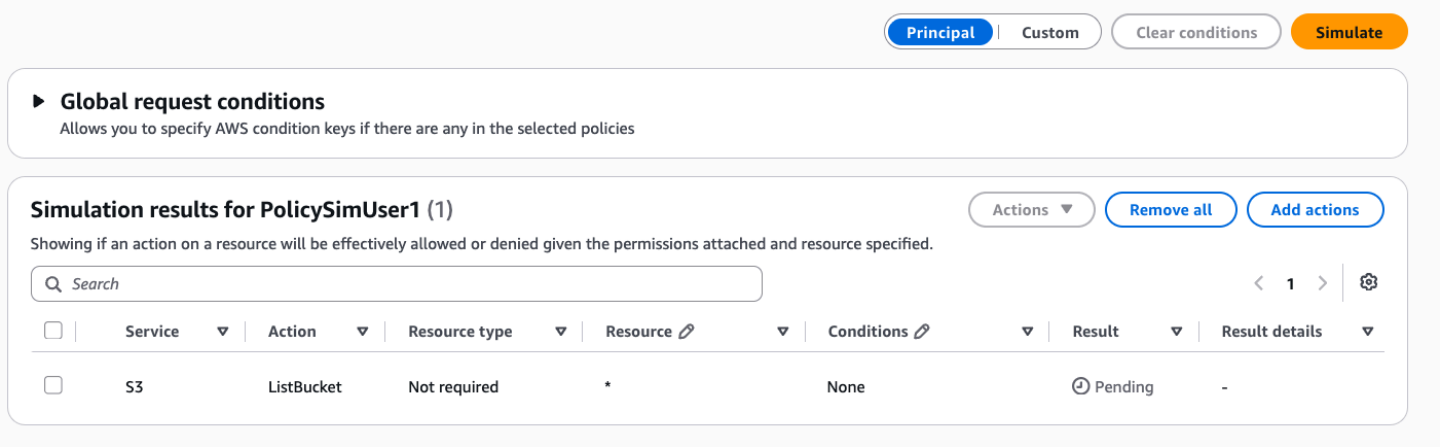
Incluir ou excluir políticas

O painel lateral lista as políticas no escopo da simulação, incluindo políticas em linha, gerenciadas pela AWS e gerenciadas pelo cliente, além de qualquer limite de permissões. Você pode excluir uma política para ver como o resultado muda sem ela e depois incluí-la novamente. A exclusão de uma política que concedeu acesso altera o resultado para uma negação implícita.

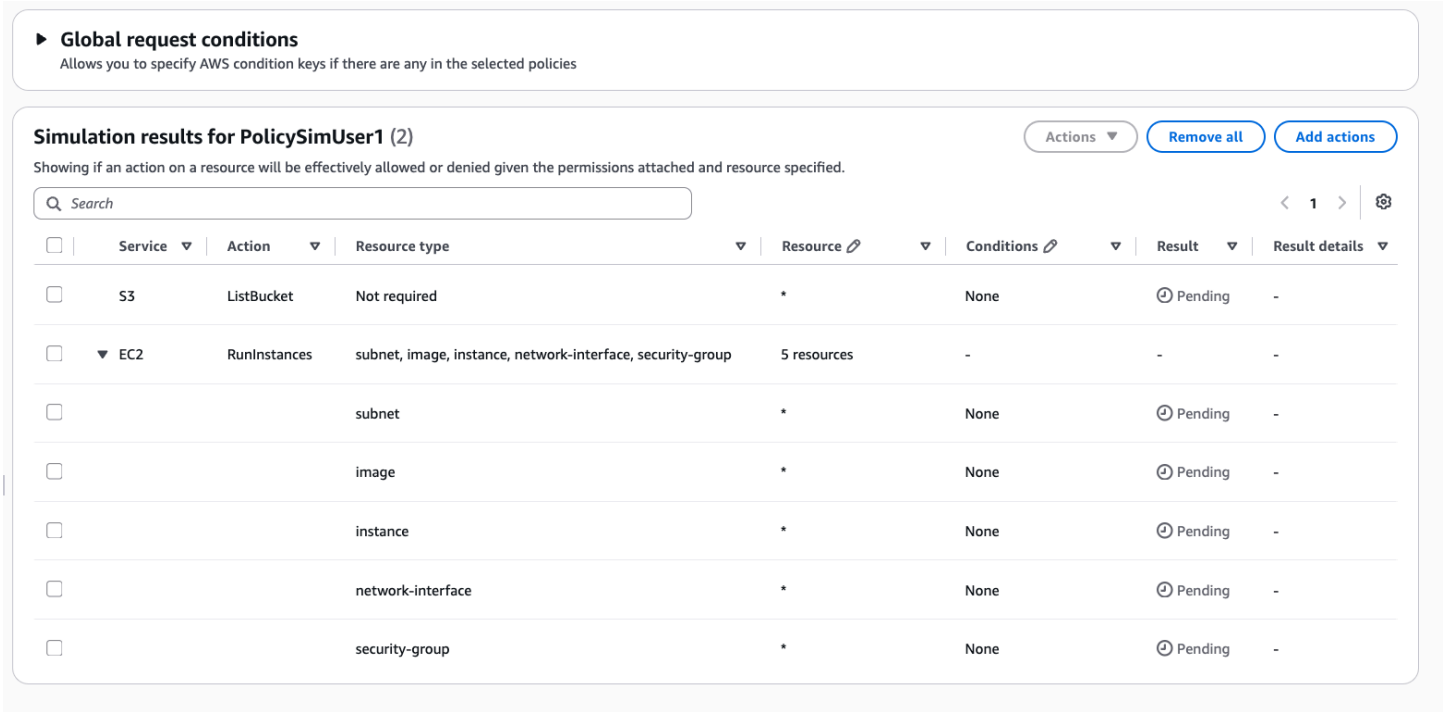
Adicionar ações e recursos



Depois de escolher identidades e políticas no painel esquerdo, adicione as ações que você deseja testar. Para uma ação que ofereça suporte a uma seleção de recursos, você pode testar todos os recursos ou inserir um ou mais ARNs de recursos específicos. Quando várias ações usam o mesmo tipo de recurso, você pode aplicar um recurso a todas elas de uma vez ou aplicar o escopo a uma única ação.



Algumas ações exigem um grupo de tipos de recursos relacionados antes que o simulador possa avaliá-las. Por exemplo, quando você simula `ec2:RunInstances`, o console solicita que você especifique uma imagem, uma instância e um grupo de segurança juntos, além de uma sub-rede ou um volume, dependendo do cenário.



Forneça valores de chave de condição

Se as políticas testadas incluírem chaves de condição, o simulador listará as chaves relevantes e permitirá que você defina valores antes de executar a simulação. As chaves mostradas refletem as políticas atualmente selecionadas; portanto, selecionar ou limpar uma política adiciona ou remove suas chaves. Se um valor não atender a uma condição, a ação será negada.

O simulador fornece valores para chaves de condição em dois lugares. As condições de solicitação global se aplicam a toda a solicitação simulada. Essas são as chaves que começam com `aws:`, como `aws:MultiFactorAuthPresent` ou `aws:PrincipalServiceNamesList`, e você define seus valores uma vez na área de condições globais. Selecionar ou limpar uma política adiciona ou remove suas chaves globais, e os valores inseridos são mantidos à medida que você ajusta quais políticas estão no escopo. As condições específicas do serviço se aplicam a uma única ação. Cada linha de ação mostra o número de condições a que ela faz referência, e você abre a linha para definir as chaves específicas do serviço para essa ação, como `cloudwatch:namespace` para `cloudwatch:PutMetricData` ou `ram:PermissionResourceType` para `ram:ListPermissionAssociations`. Uma linha de

ação pode incluir uma chave global e uma chave específica do serviço e, se algum valor não atender a uma condição, essa ação será negada.

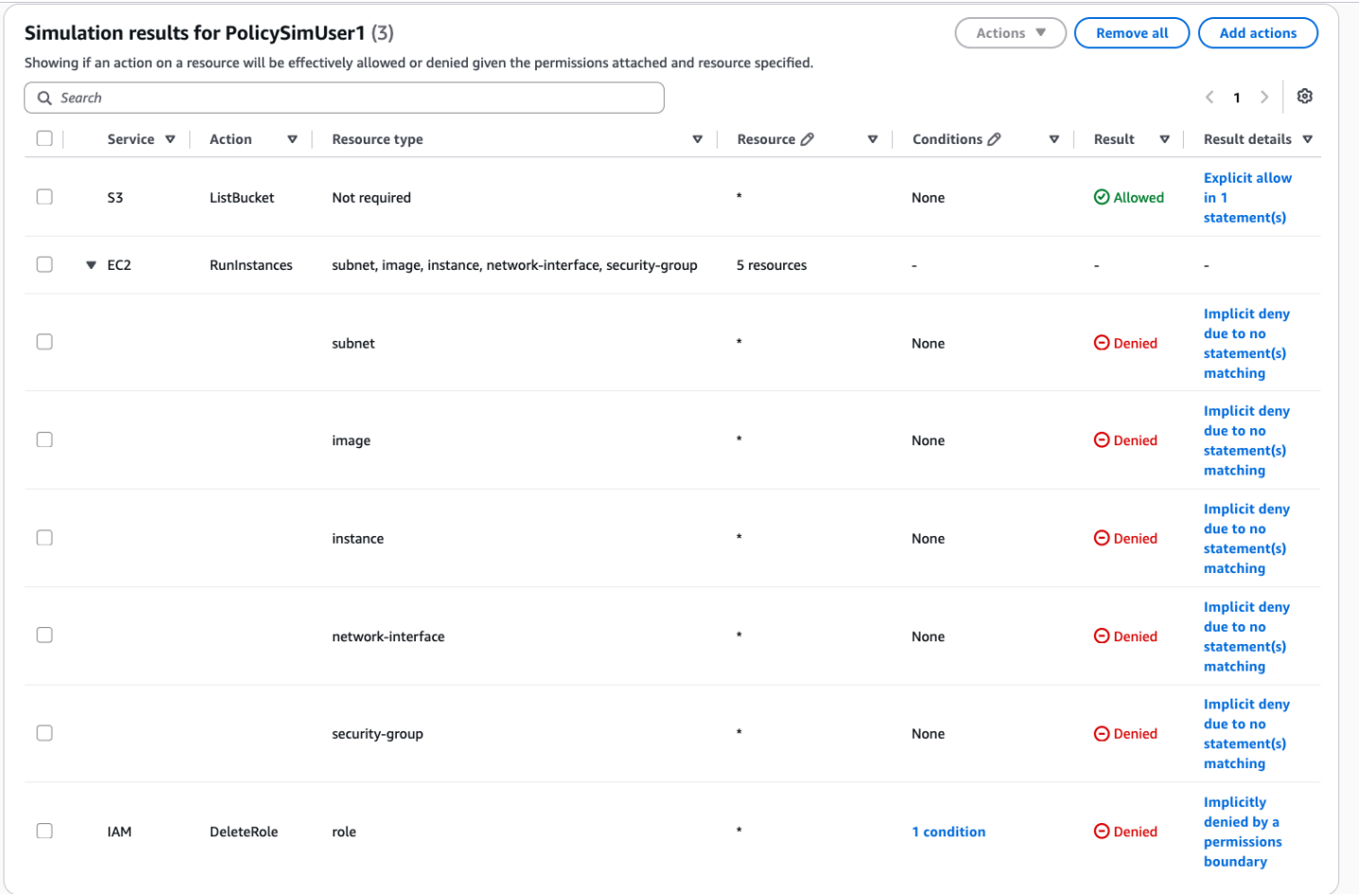
O simulador de políticas do IAM preenche algumas chaves de contexto de entidade principal e organizacional para você, e você fornece valores para o restante. Para obter a lista de chaves que são preenchidas automaticamente e mais informações sobre como as chaves de condição são avaliadas, consulte Como funciona a avaliação da chave de condição no simulador de políticas do IAM.

Para saber como usar chaves de condição, consulte AWSChaves de contexto de condições globais da.

Testar um limite de permissões

Quando uma identidade tem um limite de permissões, ela aparece no painel lateral e é incluída por padrão. Você também pode excluir o limite e executar a simulação novamente, se quiser descobrir se o limite é o motivo pelo qual uma ação foi negada.

Por exemplo, vamos supor que você simule `iam:DeleteRole` para uma identidade cujo limite não permite isso. O resultado é uma negação atribuída ao limite de permissões. Se você adicionar uma política que permita a ação e excluir o limite, o resultado será alterado para uma permissão. Isso confirma que o limite foi o único controle que bloqueou a ação.



Incluir políticas de controle de serviço

Se a sua conta for membro de uma organização, você poderá incluir políticas de controle de serviço (SCPs) na simulação no painel lateral. O simulador avalia as chaves de condição preenchidas automaticamente e o escopo dos recursos em SCPs ao determinar o resultado. Quando uma SCP nega explicitamente uma ação, o resultado é uma negação explícita atribuída ao AWS Organizations, e uma permissão em outra política não pode substituí-la. Desative o controle e execute a simulação novamente para ver o resultado sem os controles da organização. Para obter uma lista de chaves de condição preenchidas automaticamente, consulte Como funciona a avaliação da chave de condição no simulador de políticas do IAM.

> **Note**
>
> Por motivos de segurança, o simulador não revela os detalhes da avaliação do SCP. As chaves de condição referenciadas somente por SCPs não são listadas para que você defina valores, e as instruções correspondentes retornadas para um resultado não incluem

instruções de SCPs. As chaves e as instruções correspondentes vêm das políticas baseadas em identidade anexadas à identidade e de quaisquer políticas que você fornecer.

Testar uma política baseada em recurso

Para incluir uma política baseada em recursos em uma simulação de console, insira um ARN de recurso específico e escolha a opção de incluir a política do recurso. A política baseada em recursos aparece no painel lateral e é avaliada no resultado. No console, isso está disponível para buckets do Amazon S3, filas do Amazon SQS, tópicos do Amazon SNS e cofres do Amazon Glacier.

> **Note**
>
> A simulação de políticas baseadas em recursos não é compatível com perfis do IAM.

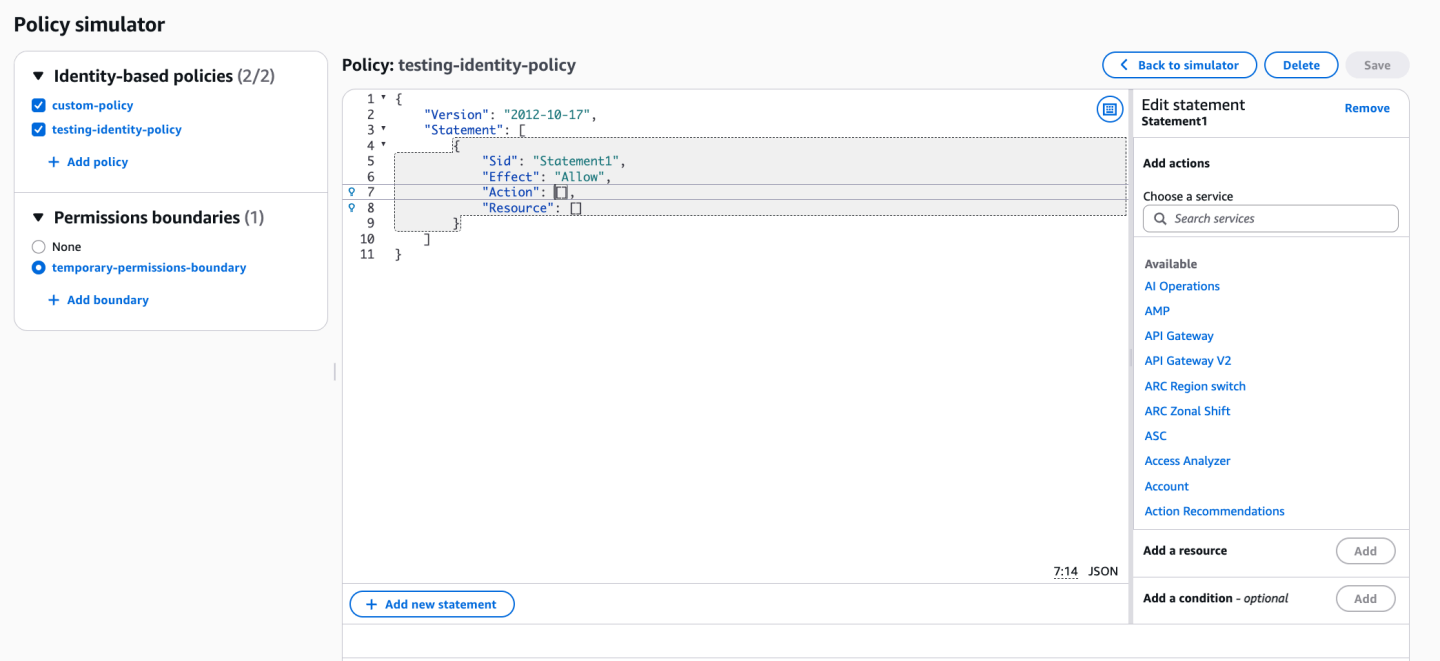
Ler os resultados

Cada linha na tabela de resultados mostra se a ação é permitida, negada implicitamente ou negada explicitamente, junto com uma breve explicação do resultado. Use a explicação para identificar qual política ou controle determinou o resultado, como uma permissão explícita, a ausência de uma instrução correspondente, um limite de permissões ou um controle organizacional.

Modo personalizado

Você pode usar o editor de políticas do IAM para escrever ou colar políticas baseadas em identidade e limites de permissões no modo personalizado. Essas políticas não estão associadas a uma identidade e podem ser avaliadas antes de você aplicá-las. Você pode incluir várias políticas baseadas em identidade, mas apenas um limite de permissões para simulação.

> **Note**
>
> No modo Personalizado, o console não oferece suporte à transmissão de SCPs personalizadas nem à simulação de políticas baseadas em recursos. Para simular SCPs personalizados ou políticas baseadas em recursos, use a AWS CLI ou a API da AWS.



Simular políticas (AWS CLI e API da AWS)

A simulação de políticas com a AWS CLI ou API da AWS geralmente envolve duas etapas:

1. Avaliar as políticas e retornar a lista de chaves de contexto às quais elas fazem referência, para que você saiba quais valores de chave de contexto fornecer. Esta etapa é opcional.
2. Simular as políticas fornecendo as ações, recursos e valores de chave de contexto a serem usados durante a simulação.

Cada ação é combinada com cada recurso, e a simulação determina se as políticas permitem ou negam essa ação para esse recurso. Você pode fornecer valores para qualquer chave de contexto referenciada por suas políticas. Se você não fornecer um valor para uma chave de contexto, a simulação ainda será executada e retornará uma negação implícita, e as chaves ausentes serão retornadas em `missingContextValues` para que você saiba quais valores adicionar. Se uma chave de contexto ausente for referenciada somente por um SCP, essa chave não será retornada por motivos de segurança, mas você ainda receberá a decisão.

As ações da API são divididas em dois grupos:

- Ações que simulam apenas as políticas passadas diretamente para a API como strings: GetContextKeysForCustomPolicy e SimulateCustomPolicy.

- Ações que simulam as políticas anexadas a um usuário, grupo de usuários, perfil ou recurso do IAM especificado: GetContextKeysForPrincipalPolicy e SimulatePrincipalPolicy. Como podem revelar permissões atribuídas a outras entidades do IAM, considere restringir o acesso a elas.

Essas ações também são compatíveis com o seguinte:

- `OrderedOrganizationPolicyInputList` em SimulateCustomPolicy permite simular o efeito da hierarquia de SCP de uma organização sem usar uma entidade principal ativa.
- `PolicyExclusionList` em SimulatePrincipalPolicy permite testar cenários “e se eu removesse essa política?” sem alterar quais políticas estão associadas à entidade principal.
- A resposta retorna um único `EvaluationResult` por ação, com decisões agregadas no nível superior e detalhes por recurso em `ResourceSpecificResults`.

Referência da AWS CLI e API da AWS

Para exemplos de comandos da AWS CLI para simular políticas, consulte o seguinte:

- AWS CLI: get-context-keys-for-custom-policy, get-context-keys-for-principal-policy, simulate-custom-policy, simulate-principal-policy
- API da AWS: GetContextKeysForCustomPolicy, GetContextKeysForPrincipalPolicy, SimulateCustomPolicy, SimulatePrincipalPolicy

Adicionar e remover permissões de identidade do IAM

Use políticas para definir as permissões para uma identidade (usuário, grupo de usuários ou função). Você pode adicionar e remover permissões anexando e desvinculando políticas do IAM em uma identidade usando o Console de gerenciamento da AWS, o AWS Command Line Interface (AWS CLI) ou a API da AWS. Você também pode usar políticas para definir limites de permissões apenas para as entidades (usuários ou perfis) que estejam usando os mesmos métodos. Os limites de permissões são um recurso da AWS avançado que controla o número máximo de permissões que uma entidade pode ter.

Tópicos

- Terminologia
- Visualizar atividade da identidade
- Adicionar permissões de identidade do IAM (console)

- Remover permissões de identidade do IAM (console)
- Adição de políticas do IAM (AWS CLI)
- Remoção de políticas do IAM (AWS CLI)
- Adição de políticas do IAM (API da AWS)
- Remoção de políticas do IAM (API da AWS)

Terminologia

Quando você associa políticas de permissões a identidades (usuários, grupos e perfis do IAM), a terminologia e os procedimentos variam dependendo de estar trabalhando com uma política gerenciada ou uma política em linha:

- Anexar: usada com políticas gerenciadas. Você anexa uma política gerenciada a uma identidade (usuário, grupo de usuários ou função). A anexação de uma política aplica as permissões da política à identidade.
- Desvincular: usada com políticas gerenciadas. Você desvincula uma política gerenciada de uma identidade do IAM (usuário, grupo de usuários ou função). A desanexação de uma política remove as permissões da identidade.
- Incorporar: usada com políticas em linha. Você incorpora uma política em linha em uma identidade (usuário, grupo de usuários ou função). A incorporação de uma política aplica as permissões da política na identidade. Como uma política em linha é armazenada na identidade, ela é incorporada em vez de anexada, embora os resultados sejam semelhantes.

  > **Note**
  >
  > É possível incorporar uma política em linha para uma função vinculada ao serviço somente no serviço que depende da função. Consulte a Documentação da AWS do seu serviço para saber se é compatível com esse recurso.

- Excluir: usada com políticas em linha. Você exclui uma política em linha de uma identidade (usuário, grupo de usuários ou função) do IAM. A exclusão de uma política remove as permissões da identidade.

> **Note**
>
> É possível excluir uma política em linha de uma função vinculada ao serviço somente no serviço que depende da função. Consulte a Documentação da AWS do seu serviço para saber se é compatível com esse recurso.

Você pode usar o console, a AWS CLI ou a API da AWS para realizar qualquer uma dessas ações.

Mais informações

- Para obter mais informações sobre a diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.
- Para obter mais informações sobre esses limites de permissões, consulte Limites de permissões para entidades do IAM.
- Para obter informações gerais sobre políticas do IAM, consulte Políticas e permissões no AWS Identity and Access Management.
- Para obter informações sobre como validar as políticas do IAM, consulte Validação de política do IAM.
- O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Visualizar atividade da identidade

Antes de alterar as permissões para uma identidade (usuário, grupo de usuários ou função), você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Adicionar permissões de identidade do IAM (console)

Você pode usar a Console de gerenciamento da AWS para adicionar permissões a uma identidade (usuário, grupo de usuários ou função). Para isso, anexe políticas gerenciadas que controlem permissões ou especifique uma política que sirva como um limite de permissões. Você também pode incorporar uma política em linha.

Para usar uma política gerenciada como uma política de permissões para uma identidade (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, selecione o botão de seleção ao lado do nome da política a ser anexada. Você pode usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha Actions (Ações) e Attach (Anexar).
5. Selecione uma ou mais identidades às quais deseja anexar a política. Você pode usar a caixa de pesquisa para filtrar a lista de entidades de segurança. Depois de selecionar as identidades, escolha Anexar política.

Para usar uma política gerenciada para definir um limite de permissões (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política a ser definida. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Na página de detalhes da política, escolha a guia Entidades anexadas e, se necessário, abra a seção Anexadas como limites de permissões e escolha Definir essa política como um limite de permissões.
5. Selecione um ou mais usuários ou funções nos quais usar a política para um limite de permissões. Você pode usar a caixa de pesquisa para filtrar a lista de entidades de segurança. Após selecionar as entidades principais, escolha Definir limite de permissões.

Para incorporar uma política em linha de um usuário ou uma função (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Usuários ou Funções.
3. Na lista, selecione o nome do grupo, do usuário ou da função para incorporar uma política.
4. Escolha a aba Permissões.

5. Escolha Adicionar permissões e depois Criar política em linha.

> **Note**
>
> Você não pode incorporar uma política em linha em uma função vinculada ao serviço no IAM. Como o serviço vinculado determina se as permissões da função podem ou não ser modificadas, você pode adicionar políticas adicionais do console de serviço, da API ou da AWS CLI. Para visualizar a documentação da função vinculada ao serviço para um serviço, consulte AWSServiços da que funcionam com o IAM e selecione Sim na coluna Função vinculada ao serviço para o seu serviço.

6. Selecione um dos seguintes métodos para visualizar as etapas necessárias para criar a política:

   - Importar políticas gerenciadas existentes: você pode importar uma política gerenciada para sua conta e, em seguida, editá-la para personalizá-la de acordo com seus requisitos específicos. Uma política gerenciada pode ser uma política gerenciada pela AWS ou uma política gerenciada pelo cliente que tenha sido criada anteriormente.
   - Criar políticas com o editor visual: você pode criar uma nova política do zero no editor visual. Se você usar o editor visual, não precisará entender a sintaxe JSON.
   - Criar políticas usando o editor de JSON: na opção de editor JSON, você pode usar a sintaxe JSON para criar uma política. Você pode digitar um novo documento de política JSON ou colar um exemplo de política.

7. Após criar uma política em linha, ela é automaticamente incorporada ao seu usuário ou função.

Para incorporar uma política em linha a um grupo de usuários (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione User groups (Grupos de usuários).
3. Na lista, escolha o nome do grupo de usuários no qual deseja incorporar uma política.
4. Escolha a guia Permissions (Permissões), Add permissions (Adicionar permissões) e Create inline policy (Criar política em linha).
5. Execute um destes procedimentos:

   - Escolha a opção Visual para criar a política. Para obter mais informações, consulte Criar políticas com o editor visual.

- Escolha a opção JSON para criar a política. Para obter mais informações, consulte Criar políticas usando o editor de JSON.

6. Quando estiver satisfeito com a política, escolha Criar política.

Para alterar o limite de permissões para uma ou mais entidades (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política a ser definida. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Na página de detalhes da política, escolha a guia Entidades anexadas e, se necessário, abra a seção Anexada como limite de permissões. Marque a caixa de seleção ao lado dos usuários ou perfis cujos limites você deseja alterar e depois escolha Alterar.
5. Selecione uma nova política para usar para um limite de permissões. É possível usar a caixa de pesquisa para filtrar a lista de políticas. Após selecionar a política, escolha Definir limite de permissões.

Remover permissões de identidade do IAM (console)

Você pode usar o Console de gerenciamento da AWS para remover permissões de uma identidade (usuário, grupo de usuários ou função). Para isso, desanexe políticas gerenciadas que controlem permissões ou remova uma política que sirva como um limite de permissões. Você também pode excluir uma política em linha.

Para desanexar uma política gerenciada usada como uma política de permissões (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, marque o botão de seleção ao lado do nome da política a ser desanexada. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha Actions (Ações) e Detach (Desvincular).

5. Selecione as identidades das quais deseja desanexar a política. Você pode usar a caixa de pesquisa para filtrar a lista de identidades. Depois de selecionar as identidades, escolha Desanexar política.

Para remover um limite de permissões (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política a ser definida. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Na página de resumo da política, escolha a guia Entidades anexadas e, se necessário, abra a seção Anexadas como limite de permissões e escolha as entidades das quais o limite de permissões será removido. Em seguida, escolha Remover limite.
5. Confirme que você deseja remover o limite e escolha Remover limite.

Para excluir uma política em linha (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usários), Users (Usuários) ou Roles (Funções).
3. Na lista, selecione o nome do grupo de usuários, usuário ou função que tem a política que você deseja remover.
4. Escolha a aba Permissões.
5. Marque a caixa de seleção ao lado da política e escolha Remover.
6. Escolha Remover na caixa de confirmação.

Adição de políticas do IAM (AWS CLI)

Você pode usar a AWS CLI para adicionar permissões a uma identidade (usuário, grupo de usuários ou função). Para isso, anexe políticas gerenciadas que controlem permissões ou especifique uma

política que sirva como um limite de permissões. Você também pode incorporar uma política em linha.

Para usar uma política gerenciada como uma política de permissões para uma entidade (AWS CLI)

1. (Opcional) Para visualizar as informações sobre uma política gerenciada, execute os comandos a seguir:
   - Para listar políticas gerenciadas: aws iam list-policies
   - Para recuperar informações detalhadas sobre uma política gerenciada: get-policy
2. Para anexar uma política gerenciada a uma identidade (usuário, grupo de usuários ou função), use um dos seguintes comandos:
   - aws iam attach-user-policy
   - aws iam attach-group-policy
   - aws iam attach-role-policy

Para usar uma política gerenciada para definir um limite de permissões (AWS CLI)

1. (Opcional) Para visualizar as informações sobre uma política gerenciada, execute os comandos a seguir:
   - Para listar políticas gerenciadas: aws iam list-policies
   - Para recuperar informações detalhadas sobre uma política gerenciada: aws iam get-policy
2. Para usar uma política gerenciada a fim de definir o limite de permissões para uma entidade (usuário ou função), use um dos seguintes comandos:
   - aws iam put-user-permissions-boundary
   - aws iam put-role-permissions-boundary

Para incorporar uma política em linha (AWS CLI)

Para incorporar uma política em linha a uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), use um dos seguintes comandos:

- aws iam put-user-policy
- aws iam put-group-policy

- aws iam put-role-policy

Remoção de políticas do IAM (AWS CLI)

Você pode usar a AWS CLI para desanexar políticas gerenciadas que controlem permissões ou remova uma política que serve como um limite de permissões. Você também pode excluir uma política em linha.

Para desanexar uma política gerenciada usada como uma política de permissões (AWS CLI)

1. (Opcional) Para visualizar as informações sobre uma política, execute os comandos a seguir:
   - Para listar políticas gerenciadas: aws iam list-policies
   - Para recuperar informações detalhadas sobre uma política gerenciada: aws iam get-policy
2. (Opcional) Para saber mais sobre as relações entre as políticas e as identidades, execute os seguintes comandos:
   - Para listar as identidades (usuários, grupos e perfis do IAM) às quais uma política gerenciada está anexada:
     - aws iam list-entities-for-policy
   - Para listar as políticas gerenciadas anexadas a uma identidade (usuário, grupo de usuários ou função), use um dos seguintes comandos:
     - aws iam list-attached-user-policies
     - aws iam list-attached-group-policies
     - aws iam list-attached-role-policies
3. Para desvincular uma política gerenciada de uma identidade (usuário, grupo de usuários ou função), use um dos seguintes comandos:
   - aws iam detach-user-policy
   - aws iam detach-group-policy
   - aws iam detach-role-policy

Para remover um limite de permissões (AWS CLI)

1. (Opcional) Para visualizar qual política gerenciada é usada atualmente para definir o limite de permissões para um usuário ou função, execute os seguintes comandos:

- aws iam get-user
- aws iam get-role

2. (Opcional) Para visualizar os usuários ou funções nos quais uma política gerenciada é usada para um limite de permissões, execute o seguinte comando:

   - aws iam list-entities-for-policy

3. (Opcional) Para visualizar as informações sobre uma política gerenciada, execute os comandos a seguir:

   - Para listar políticas gerenciadas: aws iam list-policies
   - Para recuperar informações detalhadas sobre uma política gerenciada: aws iam get-policy

4. Para remover um limite de permissões de um usuário ou função, use um dos seguintes comandos:

   - aws iam delete-user-permissions-boundary
   - aws iam delete-role-permissions-boundary

Para excluir uma política em linha (AWS CLI)

1. (Opcional) Para listar todas as políticas em linha anexadas a uma identidade (usuário, grupo de usuários, função), use um dos seguintes comandos:

   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies

2. (Opcional) Para recuperar um documento de política em linha incorporado em uma identidade (usuário, grupo de usuários ou função), use um dos seguintes comandos:

   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy

3. Para excluir uma política em linha de uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), use um dos seguintes comandos:

   - aws iam delete-user-policy

- aws iam delete-group-policy
- aws iam delete-role-policy

Adição de políticas do IAM (API da AWS)

Você pode usar a API da AWS para anexar políticas gerenciadas que controlem permissões ou especifique uma política que sirva como um limite de permissões. Você também pode incorporar uma política em linha.

Para usar uma política gerenciada como uma política de permissões para uma entidade (API da AWS)

1. (Opcional) Para visualizar as informações sobre uma política, chame as operações a seguir:
   - Para listar políticas gerenciadas: ListPolicies
   - Para recuperar informações detalhadas sobre uma política gerenciada: GetPolicy
2. Para anexar uma política gerenciada a uma identidade (usuário, grupo de usuários ou função), chame uma das seguintes operações:
   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

Para usar uma política gerenciada para definir um limite de permissões (API da AWS)

1. (Opcional) Para visualizar as informações sobre uma política gerenciada, chame as operações a seguir:
   - Para listar políticas gerenciadas: ListPolicies
   - Para recuperar informações detalhadas sobre uma política gerenciada: GetPolicy
2. Para usar uma política gerenciada a fim de definir o limite de permissões para uma entidade (usuário ou função), chame uma das seguintes operações:
   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

Para incorporar uma política em linha (API da AWS)

Para incorporar uma política em linha a uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), chame uma das seguintes operações:

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

Remoção de políticas do IAM (API da AWS)

Você pode usar a API do AWS para desanexar políticas gerenciadas que controlem permissões ou remova uma política que serve como um limite de permissões. Você também pode excluir uma política em linha.

Para desanexar uma política gerenciada usada como uma política de permissões (API da AWS)

1. (Opcional) Para visualizar as informações sobre uma política, chame as operações a seguir:
   - Para listar políticas gerenciadas: ListPolicies
   - Para recuperar informações detalhadas sobre uma política gerenciada: GetPolicy
2. (Opcional) Para saber mais sobre as relações entre as políticas e as identidades, chame as seguintes operações:
   - Para listar as identidades (usuários, grupos e perfis do IAM) às quais uma política gerenciada está anexada:
     - ListEntitiesForPolicy
   - Para listar as políticas gerenciadas anexadas a uma identidade (usuário, grupo de usuários ou função), chame uma das seguintes operações:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Para desvincular uma política gerenciada de uma identidade (usuário, grupo de usuários ou função), chame uma das seguintes operações:
   - DetachUserPolicy
   - DetachGroupPolicy

- DetachRolePolicy

Para remover um limite de permissões (API da AWS)

1. (Opcional) Para visualizar qual política gerenciada é usada atualmente para definir o limite de permissões para um usuário ou função, chame as seguintes operações:
   - GetUser
   - GetRole
2. (Opcional) Para visualizar os usuários ou funções nos quais uma política gerenciada é usada para um limite de permissões, chame a seguinte operação:
   - ListEntitiesForPolicy
3. (Opcional) Para visualizar as informações sobre uma política gerenciada, chame as operações a seguir:
   - Para listar políticas gerenciadas: ListPolicies
   - Para recuperar informações detalhadas sobre uma política gerenciada: GetPolicy
4. Para remover um limite de permissões de um usuário ou função, chame uma das seguintes operações:
   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

Para excluir uma política em linha (API da AWS)

1. (Opcional) Para listar todas as políticas em linha anexadas a uma identidade (usuário, grupo de usuários, função), chame uma das seguintes operações:
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Opcional) Para recuperar um documento de política em linha incorporado em uma identidade (usuário, grupo de usuários ou função), chame uma das seguintes operações:
   - GetUserPolicy

- GetGroupPolicy
- GetRolePolicy

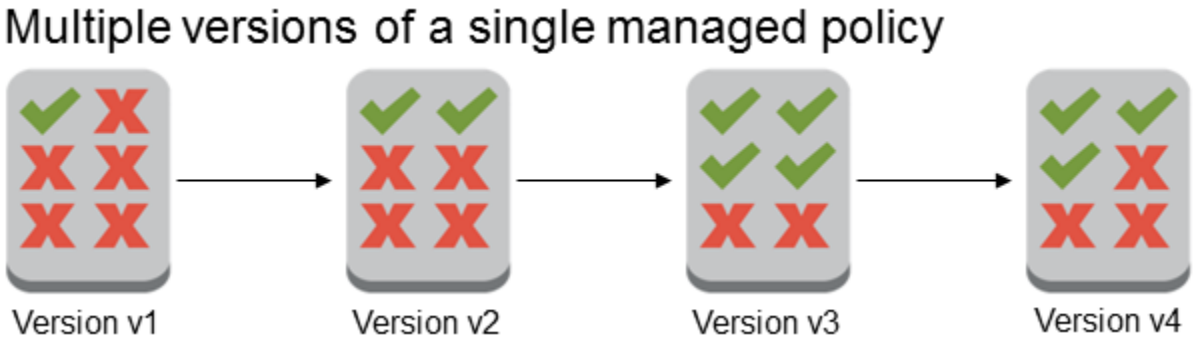
3. Para excluir uma política em linha de uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), chame uma das seguintes operações:

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Versionamento de políticas do IAM

Quando você faz alterações em uma política gerenciada pelo cliente do IAM, e quando a AWS faz alterações em uma política gerenciada pela AWS, a política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. O IAM armazena até cinco versões de suas políticas gerenciadas pelo cliente. O IAM não oferece suporte ao versionamento para políticas em linha.

O diagrama a seguir ilustra o versionamento de uma política gerenciada pelo cliente. Neste exemplo, as versões de 1 a 4 foram salvas. Você pode ter até cinco versões de políticas gerenciadas salvas no IAM. Quando você edita uma política que cria uma sexta versão salva, você pode escolher qual versão mais antiga não deve mais ser salva. Você pode reverter para qualquer uma das outras quatro versões salvas a qualquer momento.



Uma versão de política é diferente de um elemento de política `Version`. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. Para saber mais sobre o elemento de política `Version`, consulte Elementos de política JSON do IAM: Version.

Você pode usar versões para acompanhar alterações em uma política gerenciada. Por exemplo, você pode alterar uma política gerenciada e, em seguida, descobrir que a alteração teve efeitos

indesejados. Nesse caso, é possível reverter para uma versão anterior da política gerenciada definindo a versão anterior como a versão padrão.

Os tópicos a seguir explicam como usar o versionamento para políticas gerenciadas.

Tópicos

- Limites de versões
- Usar versões para reverter alterações
- Permissões para definir a versão padrão de uma política
- Definir a versão padrão de políticas gerenciadas pelo cliente

Limites de versões

Uma política gerenciada pode ter até cinco versões. Se precisar fazer alterações na política gerenciada além das cinco versões da AWS Command Line Interface ou da API do AWS, primeiro você deve excluir uma ou mais versões existentes. Se você usar o Console de gerenciamento da AWS, não será necessário excluir uma versão antes de editar sua política. Quando você salva uma sexta versão, uma caixa de diálogo é exibida solicitando que você exclua uma ou mais versões não padrão da sua política. Visualize o documento de política JSON de cada versão para ajudá-lo a decidir. Para obter detalhes sobre essa caixa de diálogo, consulte the section called “Editar políticas do IAM”.

Você pode excluir qualquer versão da política gerenciada que desejar, exceto para a versão padrão. Quando você exclui uma versão, os identificadores de versão das versões restantes não são alterados. Como resultado, os identificadores da versão talvez não sejam sequenciais. Por exemplo, se você excluir versões v2 e v4 de uma política gerenciada e adicionar duas novas versões, os identificadores das versões restantes podem ser v1, v3, v5, v6 e v7.

Usar versões para reverter alterações

Você pode definir a versão padrão de uma política gerenciada pelo cliente para reverter suas alterações. Por exemplo, considere os seguintes cenários:

Você cria uma política gerenciada pelo cliente que permite que os usuários administrem um determinado bucket do Amazon S3 usando o Console de gerenciamento da AWS. Após a criação, a política gerenciada pelo cliente tem apenas uma versão, identificada como v1, portanto, essa versão é automaticamente definida como a padrão. A política funciona como previsto.

Posteriormente, você atualiza a política para adicionar permissão para administrar um segundo bucket do Amazon S3. O IAM cria uma nova versão da política, identificada como v2, que contém as alterações. Você define a versão v2 como padrão, e pouco tempo depois seus usuários relatam que eles não têm permissão para usar o console do Amazon S3. Nesse caso, você pode reverter para a versão v1 da política, que funciona como previsto. Para fazer isso, defina a versão v1 como a versão padrão. Seus usuários agora podem usar o console do Amazon S3 para administrar o bucket original.

Depois de determinar o erro na versão v2 da política, atualize a política novamente para adicionar permissão para administrar o segundo bucket do Amazon S3. O IAM cria outra nova versão da política, identificada como v3. Você define a versão v3 como padrão, e essa versão funciona como previsto. Nesse ponto, exclua a versão v2 da política.

Permissões para definir a versão padrão de uma política

As permissões que são necessárias para definir a versão padrão de uma política correspondem às operações de API da AWS para a tarefa. Você pode usar as operações `CreatePolicyVersion` ou `SetDefaultPolicyVersion` de API para definir a versão padrão de uma política. Para permitir que algum usuário defina a versão padrão de uma política existente, você pode permitir acesso a ação `iam:CreatePolicyVersion` ou a `iam:SetDefaultPolicyVersion`. A ação `iam:CreatePolicyVersion` permite criar uma nova versão da política e definir essa versão como padrão. A ação `iam:SetDefaultPolicyVersion` permite definir qualquer versão existente da política como padrão.

> **Important**
>
> Para impedir que um usuário faça alterações na versão padrão de uma política, você deve negar `iam:CreatePolicyVersion` e `iam:SetDefaultPolicyVersion`.

Você pode usar a seguinte política para negar a um usuário acesso para alterar uma política existente gerenciada pelo cliente:

JSON

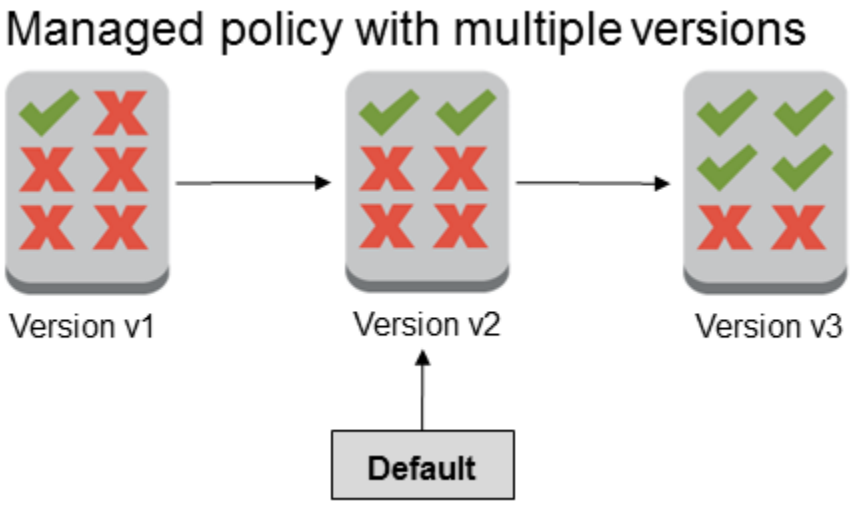
```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
}
```

Definir a versão padrão de políticas gerenciadas pelo cliente

Uma das versões de uma política gerenciada é definida como a versão padrão. A versão padrão da política é a versão operacional, ou seja, é a versão que está em vigor para todas as entidades principais (usuários do IAM, grupos do IAM e perfis do IAM) às quais a política gerenciada está anexada.

Quando você cria uma política gerenciada pelo cliente, a política começa com uma única versão identificada como v1. Para políticas gerenciadas com apenas uma única versão, essa versão é automaticamente definida como padrão. Para políticas gerenciadas pelo cliente com mais de uma versão, você escolhe qual versão definir como padrão. Para políticas gerenciadas pela AWS, a versão padrão é definida pela AWS. Os diagramas a seguir ilustram esse conceito.



É possível definir a versão padrão de uma política gerenciada pelo cliente para aplicar essa versão a todas as identidades (usuário, grupo de usuários e função) do IAM em que a política está anexada. Não é possível definir a versão padrão para uma política gerenciada pela AWS ou em linha.

Para definir a versão padrão de uma política gerenciada pelo cliente (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política para a qual deseja definir a versão padrão. Você pode usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha a guia Versões da política. Marque a caixa de seleção ao lado da versão que você deseja definir como a versão padrão e, em seguida, selecione Definir como padrão.

Para saber como definir a versão padrão de uma política gerenciada pelo cliente desde a AWS Command Line Interface ou da API da AWS, consulte Editar políticas do IAM (AWS CLI).

Editar políticas do IAM

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. As políticas são armazenadas na AWS como documentos JSON e são anexadas a entidades principais como políticas baseadas em identidade no IAM. Você pode anexar uma política baseada em identidade a uma entidade de segurança (ou identidade), como um grupo de usuários, usuário ou função do IAM. As políticas baseadas em identidade incluem as políticas gerenciadas pela AWS, as políticas gerenciadas pelo cliente e as políticas em linha. Você pode editar as políticas

gerenciadas pelo cliente e as políticas em linha no IAM. As políticas gerenciadas pela AWS não podem ser editadas. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para ter mais informações, consulte IAM e cotas do AWS STS.

Em geral, é melhor usar políticas gerenciadas pelo cliente em vez de políticas em linha ou políticas gerenciadas pela AWS. As políticas gerenciadas pela AWS costumam fornecer amplas permissões administrativas ou somente leitura. As políticas em linha não podem ser reutilizadas em outras identidades ou gerenciadas fora da identidade onde existem. Para maior segurança, conceda privilégio mínimo. Ele concede apenas as permissões necessárias para executar tarefas de trabalho específicas.

Quando você cria ou edita políticas do IAM, a AWS pode executar automaticamente a validação de políticas para ajudar você a criar uma política eficaz com o mínimo privilégio em mente. No Console de gerenciamento da AWS, o IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de política do IAM. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer.

É possível usar o Console de gerenciamento da AWS, a AWS CLI ou a API da AWS para excluir políticas gerenciadas pelo cliente e políticas em linha no IAM. Para obter mais informações sobre o uso de modelos do CloudFormation para adicionar ou atualizar políticas, consulte a Referência a tipos de recursos do AWS Identity and Access Management, no Guia do usuário do CloudFormation.

Tópicos

- Editar políticas do IAM (console)
- Editar políticas do IAM (AWS CLI)
- Editar políticas do IAM (API da AWS)

Editar políticas do IAM (console)

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. É possível usar a Console de gerenciamento da AWS para editar políticas gerenciadas pelo cliente e políticas em linha no IAM. As políticas gerenciadas pela AWS não podem ser editadas. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Para obter mais informações sobre a estrutura e a sintaxe da política, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Pré-requisitos

Antes de alterar as permissões de uma política, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Edição de políticas gerenciadas pelo cliente (console)

Você pode editar políticas gerenciadas pelo cliente para alterar as permissões definidas na política via Console de gerenciamento da AWS. Uma política gerenciada pelo cliente pode ter até cinco versões. Isso é importante, pois se você precisar fazer alterações em uma política gerenciada criando mais de cinco versões, o Console de gerenciamento da AWS solicitará que você decida qual versão excluir. Você também pode alterar a versão padrão ou excluir uma versão da política antes de editá-la para evitar ser solicitado. Para saber mais sobre versões do , consulte Versionamento de políticas do IAM.

Console

Para editar uma política gerenciada pelo cliente

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política a editar. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha a guia Permissões e depois escolha Editar.
5. Execute um destes procedimentos:

   - Escolha a opção Visual para alterar a política sem necessidade de compreender a sintaxe JSON. Você pode fazer alterações nos serviços, ações, recursos ou condições opcionais para cada bloco de permissões na política. Também é possível importar uma política e acrescentar permissões adicionais no final da sua política. Quando terminar de fazer alterações, escolha Avançar para continuar.

- Escolha a opção JSON para modificar a política digitando ou colando texto na caixa de texto JSON. Também é possível importar uma política e acrescentar permissões adicionais no final da sua política. Resolva os avisos de segurança, as mensagens erros ou os avisos gerais gerados durante a validação de política, e depois escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

6. Na página Revisar e salvar, revise Permissões definidas nessa política e escolha Salvar alterações para salvar seu trabalho.
7. Se a política gerenciada já tiver o máximo de cinco versões e você escolher Salvar alterações, uma caixa de diálogo será exibida. Para salvar a nova versão, a versão não padrão mais antiga da política será removida e substituída por essa nova versão. Opcionalmente, você pode definir a nova versão como a versão padrão da política.

   Escolha Salvar alterações para salvar a nova versão da política.

Definir a versão padrão de uma política gerenciada pelo cliente (console)

É possível definir uma versão padrão de uma política gerenciada pelo cliente via Console de gerenciamento da AWS. Você pode usar essa política para estabelecer uma configuração de linha de base consistente para permissões em toda a sua organização. Todos os novos anexos da política usarão esse conjunto padronizado de permissões.

Console

Para definir a versão padrão de uma política gerenciada pelo cliente (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política para a qual deseja definir a versão padrão. Você pode usar a caixa de pesquisa para filtrar a lista de políticas.

4. Escolha a guia Versões da política. Marque a caixa de seleção ao lado da versão que você deseja definir como a versão padrão e, em seguida, selecione Definir como padrão.

Excluir uma versão de uma política gerenciada pelo cliente (console)

Talvez seja necessário excluir uma versão de uma política gerenciada pelo cliente para remover permissões desatualizadas ou incorretas que não são mais necessárias ou representam possíveis riscos de segurança. Ao manter somente as versões necessárias, você ajuda a garantir que permaneça dentro do limite de cinco versões de política gerenciada, permitindo espaço para futuras atualizações e refinamentos. É possível excluir uma versão de uma política gerenciada pelo cliente via Console de gerenciamento da AWS.

Console

Para excluir uma versão de uma política gerenciada pelo cliente

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Selecione o nome da política gerenciada pelo cliente que possui uma versão que você deseja excluir. Você pode usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha a guia Versões da política. Marque a caixa de seleção ao lado da versão que você deseja excluir. Escolha Excluir.
5. Confirme que você deseja excluir a versão e, em seguida, selecione Excluir.

Edição de políticas em linha (console)

Talvez seja necessário editar uma política gerenciada pelo cliente para atualizar ou refinar as permissões concedidas, garantindo que elas permaneçam alinhadas aos requisitos de segurança dinâmicos e às necessidades de controle de acesso da sua organização. Editar permite ajustar o documento JSON da política, adicionando, modificando ou removendo ações, recursos ou condições específicas para manter o princípio do privilégio mínimo e se adaptar às mudanças em seu ambiente ou processos. Você pode editar uma política em linha no Console de gerenciamento da AWS.

Console

Para editar uma política em linha para um usuário, grupo de usuários ou perfil

1. No painel de navegação, escolha User (Usuários), User groups (Grupos de usuários) ou Roles (Funções).
2. Selecione o nome do usuário, grupo de usuários ou da função com a política que você deseja modificar. Em seguida, selecione a guia Permissões e expanda a política.
3. Para editar uma política em linha, escolha Editar política.
4. Execute um destes procedimentos:

   - Escolha a opção Visual para alterar a política sem necessidade de compreender a sintaxe JSON. Você pode fazer alterações nos serviços, ações, recursos ou condições opcionais para cada bloco de permissões na política. Também é possível importar uma política e acrescentar permissões adicionais no final da sua política. Quando terminar de fazer alterações, escolha Avançar para continuar.
   - Escolha a opção JSON para modificar a política digitando ou colando texto na caixa de texto JSON. Também é possível importar uma política e acrescentar permissões adicionais no final da sua política. Resolva os avisos de segurança, as mensagens erros ou os avisos gerais gerados durante a validação de política, e depois escolha Próximo. Para salvar as alterações sem afetar as entidades atualmente anexadas, desmarque a caixa de seleção para Salvar como versão padrão.

   > **Note**
   >
   > Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

5. Na página Revisar, revise o resumo da política e depois escolha Salvar alterações para salvar seu trabalho.

Editar políticas do IAM (AWS CLI)

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. É possível usar a AWS Command Line Interface (AWS CLI) para editar políticas gerenciadas pelo cliente e políticas em linha no IAM. As políticas gerenciadas pela AWS não podem ser editadas. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Para obter mais informações sobre a estrutura e a sintaxe da política, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Pré-requisitos

Antes de alterar as permissões de uma política, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Edição de políticas gerenciadas pelo cliente (AWS CLI)

Você pode editar uma política gerenciada pelo cliente na AWS CLI.

> **Note**
>
> Uma política gerenciada pode ter até cinco versões. Se precisar fazer alterações na política gerenciada de um cliente além das cinco versões, primeiro você deve excluir uma ou mais versões existentes.

Para editar uma política gerenciada pelo cliente (AWS CLI)

1. (Opcional) Para visualizar as informações sobre uma política, execute os comandos a seguir:
   - Para listar políticas gerenciadas: list-policies
   - Para recuperar informações detalhadas sobre uma política gerenciada: get-policy
2. (Opcional) Para saber mais sobre as relações entre as políticas e as identidades, execute os seguintes comandos:

- Para listar as identidades (usuários, grupos e perfis do IAM) às quais uma política gerenciada está anexada:
  - list-entities-for-policy
- Para listar as políticas gerenciadas anexadas a uma identidade (usuário, grupo de usuários ou função):
  - list-attached-user-policies
  - list-attached-group-policies
  - list-attached-role-policies

3. Para editar uma política gerenciada pelo cliente, execute o seguinte comando:
   - create-policy-version
4. (Opcional) Para validar uma política gerenciada pelo cliente, execute o seguinte comando do IAM Access Analyzer:
   - validate-policy

Definir a versão padrão de uma política gerenciada pelo cliente (AWS CLI)

É possível definir uma versão padrão de uma política gerenciada pelo cliente via AWS CLI.

Para definir a versão padrão de uma política gerenciada pelo cliente (AWS CLI)

1. (Opcional) Para listar políticas gerenciadas, execute o seguinte comando:
   - list-policies
2. Para definir a versão padrão de uma política gerenciada pelo cliente, execute o seguinte comando:
   - set-default-policy-version

Excluir uma versão de uma política gerenciada pelo cliente (AWS CLI)

É possível excluir uma versão de uma política gerenciada pelo cliente via AWS CLI.

Para excluir uma versão de uma política gerenciada pelo cliente (AWS CLI)

1. (Opcional) Para listar políticas gerenciadas, execute o seguinte comando:

- list-policies

2. Para excluir uma política gerenciada pelo cliente, execute o seguinte comando:

   - delete-policy-version

Editar políticas em linha (AWS CLI)

Você pode editar uma política em linha no AWS CLI.

Para editar uma política em linha (AWS CLI)

1. (Opcional) Para visualizar as informações sobre uma política, execute os comandos a seguir:

   - Para listar políticas em linha associadas a uma identidade (usuário, grupo de usuários ou perfil):
     - list-user-policies
     - list-role-policies
     - list-group-policies
   - Para recuperar informações detalhadas sobre uma política em linha:
     - get-user-policy
     - get-role-policy
     - get-group-policy

2. Para editar uma política em linha, execute o seguinte comando:

   - put-user-policy
   - put-role-policy
   - put-group-policy

3. (Opcional) Para validar uma política em linha, execute o seguinte comando do IAM Access Analyzer:

   - validate-policy

Editar políticas do IAM (API da AWS)

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. É possível usar a API da AWS para editar políticas gerenciadas pelo cliente e políticas

em linha no IAM. As políticas gerenciadas pela AWS não podem ser editadas. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

Para obter mais informações sobre a estrutura e a sintaxe da política, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Pré-requisitos

Antes de alterar as permissões de uma política, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Edição de políticas gerenciadas pelo cliente (API da AWS)

Você pode editar uma política gerenciada pelo cliente usando a API da AWS.

> **Note**
>
> Uma política gerenciada pode ter até cinco versões. Se precisar fazer alterações na política gerenciada de um cliente além das cinco versões, primeiro você deve excluir uma ou mais versões existentes.

Para editar uma política gerenciada pelo cliente (API da AWS)

1. (Opcional) Para visualizar as informações sobre uma política, chame as operações a seguir:
   - Para listar políticas gerenciadas: ListPolicies
   - Para recuperar informações detalhadas sobre uma política gerenciada: GetPolicy
2. (Opcional) Para saber mais sobre as relações entre as políticas e as identidades, chame as seguintes operações:
   - Para listar as identidades (usuários, grupos e perfis do IAM) às quais uma política gerenciada está anexada:
     - ListEntitiesForPolicy

- Para listar as políticas gerenciadas anexadas a uma identidade (usuário, grupo de usuários ou função):
  - ListAttachedUserPolicies
  - ListAttachedGroupPolicies
  - ListAttachedRolePolicies

3. Para editar uma política gerenciada pelo cliente, chame a seguinte operação:
   - CreatePolicyVersion
4. (Opcional) Para validar uma política gerenciada pelo cliente, chame a seguinte operação do IAM Access Analyzer:
   - ValidatePolicy

Definir a versão padrão de uma política gerenciada pelo cliente (API da AWS)

É possível definir uma versão padrão de uma política gerenciada pelo cliente via API da AWS.

Para definir a versão padrão de uma política gerenciada pelo cliente (API da AWS)

1. (Opcional) Para listar políticas gerenciadas, chame a seguinte operação:
   - ListPolicies
2. Para definir a versão padrão de uma política gerenciada pelo cliente, chame a seguinte operação:
   - SetDefaultPolicyVersion

Excluir uma versão de uma política gerenciada pelo cliente (API da AWS)

É possível excluir uma versão de uma política gerenciada pelo cliente via API da AWS.

Para excluir uma versão de uma política gerenciada pelo cliente (API da AWS)

1. (Opcional) Para listar políticas gerenciadas, chame a seguinte operação:
   - ListPolicies
2. Para excluir uma política gerenciada pelo cliente, chame a seguinte operação:

- DeletePolicyVersion

Editar políticas em linha (API da AWS)

É possível editar uma política em linha via API da AWS.

Para editar uma política em linha (API da AWS)

1. (Opcional) Para visualizar as informações sobre uma política em linha, chame as operações a seguir:
   - Para listar políticas em linha associadas a uma identidade (usuário, grupo de usuários ou perfil):
     - ListUserPolicies
     - ListRolePolicies
     - ListGroupPolicies
   - Para recuperar informações detalhadas sobre uma política em linha:
     - GetUserPolicy
     - GetRolePolicy
     - GetGroupPolicy
2. Para editar uma política em linha, chame as seguintes operações:
   - PutUserPolicy
   - PutRolePolicy
   - PutGroupPolicy
3. (Opcional) Para validar uma política em linha, execute a seguinte operação do IAM Access Analyzer:
   - ValidatePolicy

Excluir políticas do IAM

Você pode excluir as políticas do IAM usando o Console de gerenciamento da AWS, a AWS Command Line Interface (AWS CLI) ou a API da AWS.

> **Note**
>
> A exclusão de políticas do IAM é permanente. Depois que a política é excluída, não é possível recuperá-la.

Para obter mais informações sobre a estrutura e a sintaxe da política do IAM, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Para obter mais informações sobre a diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.

O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para ter mais informações, consulte IAM e cotas do AWS STS.

Tópicos

- Excluir políticas do IAM (console)
- Excluir políticas do IAM (AWS CLI)
- Excluir políticas do IAM (API da AWS)

Excluir políticas do IAM (console)

Você pode usar o Console de gerenciamento da AWS para excluir políticas gerenciadas pelo cliente e políticas em linha no IAM. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

> **Note**
>
> A exclusão de políticas do IAM é permanente. Depois que a política é excluída, não é possível recuperá-la.

Para obter mais informações sobre a estrutura e a sintaxe da política do IAM, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Para obter mais informações sobre a diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.

Pré-requisitos

Antes de excluir uma política, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Exclusão de políticas do IAM (console)

Talvez seja necessário excluir uma política gerenciada pelo cliente quando ela se tornar obsoleta ou não estiver mais alinhada aos requisitos de segurança e às necessidades de controle de acesso da sua organização. Ao excluir políticas desnecessárias, você reduz os possíveis riscos de segurança associados a políticas desatualizadas ou não utilizadas. Você pode excluir uma política gerenciada pelo cliente para removê-la de sua Conta da AWS. Você não pode excluir políticas gerenciadas pela AWS.

Console

Para excluir uma política gerenciada pelo cliente

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Marque o botão de seleção ao lado da política gerenciada pelo cliente que você deseja excluir. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha Actions (Ações) e, em seguida, escolha Delete (Excluir).
5. Siga as instruções para confirmar que deseja excluir a política e selecione Excluir.

Excluir políticas em linha (console)

Talvez seja necessário excluir uma política em linha quando as permissões específicas por ela concedidas não forem mais necessárias para o usuário, grupo ou perfil do IAM ao qual ela está diretamente anexada. A exclusão de políticas em linha desnecessárias ajuda a reduzir o risco de acesso não intencional, especialmente porque as políticas em linha não podem ser reutilizadas ou compartilhadas entre várias identidades, ao contrário das políticas gerenciadas. Você pode

excluir uma política em linha para removê-la da sua Conta da AWS. Você não pode excluir políticas gerenciadas pela AWS.

Console

Para editar uma política em linha para um grupo, usuário ou perfil do IAM

1. No painel de navegação, escolha User groups (Grupos de usários), Users (Usuários) ou Roles (Funções).
2. Selecione o nome do grupo de usuários, do usuário ou da função com a política que você deseja excluir. Em seguida, escolha a guia Permissões.
3. Marque as caixas de seleção ao lado das políticas que você deseja excluir e escolha Remover. Em seguida, na caixa de diálogo de confirmação, confirme a remoção e a exclusão da política.

   - Para excluir uma política em linha em Usuários ou Perfis, escolha Remover para confirmar a exclusão.
   - Se você estiver excluindo uma única política em linha em User groups (Grupos de usuários), digite o nome da política e escolha Delete (Excluir). Se você estiver excluindo várias políticas em linha em User groups (Grupos de usuários), digite o número de políticas que você está excluindo seguido de **`inline policies`** e escolha Delete (Excluir). Por exemplo, se você estiver excluindo três políticas em linha, digite **`3 inline policies`**.

Excluir políticas do IAM (AWS CLI)

Você pode usar a AWS Command Line Interface (AWS CLI) para excluir políticas gerenciadas pelo cliente e políticas em linha no IAM. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

> **Note**
>
> A exclusão de políticas do IAM é permanente. Depois que a política é excluída, não é possível recuperá-la.

Para obter mais informações sobre a estrutura e a sintaxe da política do IAM, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Para obter mais informações sobre a diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.

Pré-requisitos

Antes de excluir uma política, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Excluir políticas gerenciadas pelo cliente (AWS CLI)

Você pode excluir uma política gerenciada pelo cliente na AWS Command Line Interface.

Para excluir uma política gerenciada pelo cliente (AWS CLI)

1. (Opcional) Para visualizar as informações sobre uma política, execute os comandos a seguir:
   - Para listar políticas gerenciadas: list-policies
   - Para recuperar informações detalhadas sobre uma política gerenciada: get-policy
2. (Opcional) Para saber mais sobre as relações entre as políticas e as identidades, execute os seguintes comandos:
   - Para listar as identidades (usuários, grupos e perfis do IAM) às quais uma política gerenciada está anexada, execute o seguinte comando:
     - list-entities-for-policy
   - Para listar as políticas gerenciadas anexadas a uma identidade (usuário, grupo de usuários ou função), execute um dos seguintes comandos:
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Para excluir uma política gerenciada pelo cliente, execute o seguinte comando:
   - delete-policy

Excluir políticas em linha (AWS CLI)

É possível excluir uma política em linha via AWS CLI.

Para excluir uma política em linha (AWS CLI)

1. (Opcional) Para listar todas as políticas em linha anexadas a uma identidade (usuário, grupo de usuários, função), use um dos seguintes comandos:
   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies
2. (Opcional) Para recuperar um documento de política em linha incorporado em uma identidade (usuário, grupo de usuários ou função), use um dos seguintes comandos:
   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy
3. Para excluir uma política em linha de uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), use um dos seguintes comandos:
   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Excluir políticas do IAM (API da AWS)

Você pode usar a API da AWS para excluir políticas gerenciadas pelo cliente e políticas em linha no IAM. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

> **Note**
>
> A exclusão de políticas do IAM é permanente. Depois que a política é excluída, não é possível recuperá-la.

Para obter mais informações sobre a estrutura e a sintaxe da política do IAM, consulte Políticas e permissões no AWS Identity and Access Management e Referência de elemento de política JSON do IAM.

Para obter mais informações sobre a diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.

Pré-requisitos

Antes de excluir uma política, você deve revisar a atividade no nível de serviço recente. Isso é importante porque você não deseja remover acesso de uma entidade principal (pessoa ou aplicativo) que está usando. Para obter mais informações sobre como visualizar as informações acessadas por último, consulte Refinar permissões na AWS usando informações do último acesso.

Excluir políticas gerenciadas pelo cliente (API da AWS)

Você pode excluir uma política gerenciada pelo cliente usando a API da AWS.

Para excluir uma política gerenciada pelo cliente (API da AWS)

1. (Opcional) Para visualizar as informações sobre uma política, chame as operações a seguir:
   - Para listar políticas gerenciadas: ListPolicies
   - Para recuperar informações detalhadas sobre uma política gerenciada: GetPolicy
2. (Opcional) Para saber mais sobre as relações entre as políticas e as identidades, chame as seguintes operações:
   - Para listar as identidades (usuários, grupos e perfis do IAM) às quais uma política gerenciada está anexada, chame a seguinte operação:
     - ListEntitiesForPolicy
   - Para listar as políticas gerenciadas anexadas a uma identidade (usuário, grupo de usuários ou função), chame uma das seguintes operações:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Para excluir uma política gerenciada pelo cliente, chame a seguinte operação:
   - DeletePolicy

Excluir políticas em linha (API da AWS)

É possível excluir políticas em linha via API da AWS.

Para excluir uma política em linha (API da AWS)

1. (Opcional) Para listar todas as políticas em linha anexadas a uma identidade (usuário, grupo de usuários, função), chame uma das seguintes operações:
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Opcional) Para recuperar um documento de política em linha incorporado em uma identidade (usuário, grupo de usuários ou função), chame uma das seguintes operações:
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy
3. Para excluir uma política em linha de uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), chame uma das seguintes operações:
   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Refinar permissões na AWS usando informações do último acesso

Como administrador, é possível conceder permissões a recursos do IAM (perfis, usuários, grupos de usuários ou políticas) além do que é exigido. O IAM fornece as informações do último acesso para ajudar você a identificar as permissões não utilizadas para que você possa removê-las. É possível usar essas informações para refinar suas políticas e permitir o acesso somente aos serviços e ações usados por suas identidades e políticas do IAM. Isso ajuda você a melhor seguir as práticas recomendadas de privilégio mínimo. É possível visualizar as informações de acesso mais recente de identidades ou políticas existentes no IAM ou no AWS Organizations.

É possível monitorar continuamente as informações do último acesso com analisadores de acessos não utilizados. Para obter mais informações, consulte Findings for external and unused access.

Tópicos

- Tipos de informações do último acesso do IAM

- Informações acessadas por último do AWS Organizations
- Coisas para saber sobre as informações acessadas por último
- Permissões obrigatórias
- Atividades de solução de problemas para entidades do IAM e do AWS Organizations
- Onde a AWS rastreia informações acessadas por último
- Visualizar as últimas informações acessadas do IAM
- Visualizar as últimas informações acessadas do AWS Organizations
- Cenários de exemplo para uso de informações acessadas por último
- Serviços e ações para os quais a ação do IAM acessou informações pela última vez

Tipos de informações do último acesso do IAM

É possível visualizar dois tipos de informações de acesso mais recente de identidades do IAM: informações permitidas do serviço da AWS e informações permitidas da ação. As informações incluem a data e a hora em que a tentativa de acesso a uma API da AWS foi feita. Para ações, as informações de acesso mais recente relatam as ações de gerenciamento de serviços. As ações de gerenciamento incluem ações de criação, exclusão e modificação. Para saber mais sobre como visualizar as informações do último acesso do IAM, consulte Visualizar as últimas informações acessadas do IAM.

Para obter cenários de exemplo de uso das informações de acesso mais recente para tomar decisões sobre as permissões concedidas às entidades do IAM, consulte Cenários de exemplo para uso de informações acessadas por último.

Para saber mais sobre como as informações para ações de gerenciamento são fornecidas, consulte Coisas para saber sobre as informações acessadas por último.

Informações acessadas por último do AWS Organizations

Se você fizer login usando as credenciais da conta de gerenciamento, poderá visualizar as informações do último acesso ao serviço para uma entidade ou política do AWS Organizations em sua organização. As entidades do AWS Organizations incluem a raiz da organização, unidades organizacionais (OUs) ou contas. As informações acessadas por último para o AWS Organizations incluem informações sobre serviços que são permitidos por uma política de controle de serviço (SCP). As informações indicam quais entidades principais (usuário raiz, usuário ou perfil do IAM) de uma organização ou conta tentaram acessar o serviço por último e quando isso ocorreu. Para

saber mais sobre o relatório e como visualizar as informações acessadas por último para o AWS Organizations, consulte Visualizar as últimas informações acessadas do AWS Organizations.

Para obter cenários de exemplo para usar as informações acessadas por último para tomar decisões sobre as permissões concedidas às entidades do AWS Organizations, consulte Cenários de exemplo para uso de informações acessadas por último.

Coisas para saber sobre as informações acessadas por último

Para usar as informações acessadas por último em um relatório para alterar as permissões de uma entidade do IAM ou do AWS Organizations, examine os seguintes detalhes sobre as informações.

- Período de rastreamento: a atividade recente geralmente aparece no console do IAM em até quatro horas. O período de rastreamento para as informações de serviço é de, no mínimo, 400 dias, dependendo do momento em que o serviço começou a rastrear as informações das ações. O período de rastreamento das informações de ações do Amazon S3 começou em 12 de abril de 2020. O período de rastreamento das ações do Amazon EC2, do IAM e do Lambda começou em 7 de abril de 2021. O período de rastreamento para todos os outros serviços começou em 23 de maio de 2023. Para obter uma lista de serviços para os quais informações acessadas pela última vez pela ação estão disponíveis, consulte Serviços e ações para os quais a ação do IAM acessou informações pela última vez. Para obter mais informações sobre em quais regiões informações acessadas pela última vez pela ação estão disponíveis, consulte Onde a AWS rastreia informações acessadas por último.
- Tentativas relatadas: os dados do último acesso ao serviço incluem todas as tentativas de acesso a uma API da AWS, não somente as tentativas bem-sucedidas. Isso inclui todas as tentativas de acesso que foram feitas usando o Console de gerenciamento da AWS, a API da AWS por meio de qualquer um dos SDKs ou qualquer uma das ferramentas da linha de comando. Uma entrada inesperada nos dados de últimos serviços acessados não significa que sua conta foi comprometida, pois a solicitação pode ter sido negada. Consulte os logs do CloudTrail como a fonte segura para obter informações sobre todas as chamadas de API e se elas foram bem-sucedidas ou tiveram acesso negado.
- PassRole: a ação `iam:PassRole` não é rastreada e não está incluída nas informações acessadas pela última vez pela ação do IAM.
- Informações acessadas pela última vez pela ação: as informações acessadas pela última vez pela ação estão disponíveis para ações de gerenciamento acessadas por identidades do IAM. Veja a lista de serviços e suas ações para as quais a ação acessou informações de relatório pela última vez.

> **Note**
>
> As informações acessadas pela última vez pela ação não estão disponíveis para eventos de plano de dados.

- Eventos de gerenciamento: o IAM fornece informações de ação para eventos de gerenciamento de serviço que são registrados em log pelo CloudTrail. Às vezes, eventos de gerenciamento do CloudTrail também são chamados de operações do ambiente de gerenciamento ou eventos do ambiente de gerenciamento. Eventos de gerenciamento fornecem visibilidade nas operações de gerenciamento executadas em recursos na sua Conta da AWS. Para saber mais sobre eventos de gerenciamento no CloudTrail, consulte Registro em log de eventos de gerenciamento no Guia do usuário do AWS CloudTrail.
- Report owner (Proprietário do relatório): somente a entidade de segurança que gera um relatório pode visualizar os detalhes do relatório. Isso significa que, quando você visualizar as informações no Console de gerenciamento da AWS, talvez precisará esperar que elas sejam geradas e carregadas. Se você usar a AWS CLI ou a API da AWS para obter detalhes do relatório, suas credenciais deverão corresponder às credenciais do principal que gerou o relatório. Se você usar credenciais temporárias para um perfil ou uma entidade principal de usuário federado do AWS STS, será necessário gerar e recuperar o relatório durante a mesma sessão. Para obter mais informações sobre as entidades principais de sessão de função assumida, consulte AWS Elementos da política JSON da: Principal.
- Recursos do IAM: as informações sobre os acessos mais recentes ao IAM incluem os recursos do IAM (perfis, usuários políticas e grupos do IAM) da sua conta. As informações de acesso mais recente do AWS Organizations incluem entidades principais (usuários do IAM, perfis do IAM ou o Usuário raiz da conta da AWS) na entidade especificada do AWS Organizations. As informações de acesso mais recente não incluem tentativas não autenticadas.
- Tipos de política do IAM: as informações de acesso mais recente do IAM incluem serviços permitidos por políticas de uma identidade do IAM. Essas são as políticas anexadas a uma função ou a um usuário diretamente ou por meio de um grupo. O acesso permitido por outros tipos de política não está incluído no relatório. Os tipos de política excluídos incluem políticas baseadas em recurso, listas de controle de acesso, SCPs do AWS Organizations, limites de permissões do IAM e políticas de sessão. As permissões fornecidas por funções vinculadas ao serviço são definidas pelo serviço ao qual estão vinculadas e não podem ser modificadas no IAM. Para saber mais sobre funções vinculadas ao serviço, consulte Criar um perfil vinculado ao serviço Para saber

como os diferentes tipos de política são avaliados para permitir ou negar acesso, consulte Lógica da avaliação de política.

- Tipos de política do AWS Organizations: as informações do AWS Organizations incluem somente os serviços permitidos por políticas de controle de serviço (SCPs) herdadas de uma entidade do AWS Organizations. SCPs são políticas anexadas a uma raiz, UO ou conta. O acesso permitido por outros tipos de política não está incluído no relatório. Os tipos de política excluídos incluem políticas baseadas em identidade, políticas baseadas em recurso, listas de controle de acesso, limites de permissões do IAM e políticas de sessão. Para saber como os tipos de política diferentes são avaliados para permitir ou negar acesso, consulte Lógica da avaliação de política.
- Especificar um ID de política: ao usar a AWS CLI ou a API da AWS para gerar um relatório de informações acessadas por último no AWS Organizations, você também pode especificar um ID de política. O relatório resultante inclui informações dos serviços permitidos somente por essa política. As informações incluem as atividades de conta mais recentes na entidade do AWS Organizations especificada ou filhos da entidade. Para obter mais informações, consulte aws iam generate-organizations-access-report ou GenerateOrganizationsAccessReport.
- Conta gerencial do AWS Organizations: você deve fazer login na conta gerencial de sua organização para visualizar as informações do último acesso ao serviço. Você pode escolher visualizar informações da conta de gerenciamento usando o console do IAM, a AWS CLI, ou a API da AWS. O relatório resultante lista todos os produtos da AWS, porque a conta de gerenciamento não é limitada pelas SCPs. Se você especificar um ID de política na CLI ou API, a política será ignorada. Para cada serviço, o relatório inclui informações somente da conta de gerenciamento. No entanto, os relatórios para outras entidades do AWS Organizations não retornam informações de atividades na conta gerencial.
- Configurações do AWS Organizations: um administrador deve habilitar as SCPs na raiz de sua organização antes que você possa gerar dados do AWS Organizations.

Permissões obrigatórias

Para visualizar as informações acessadas por último no Console de gerenciamento da AWS, você deve ter uma política que conceda as permissões necessárias.

Permissões para visualizar informações do IAM

Para usar o console do IAM para visualizar as informações do último acesso de um usuário, uma função ou uma política do IAM, você deve ter uma política que inclua as seguintes ações:

- `iam:GenerateServiceLastAccessedDetails`

- `iam:Get*`
- `iam:List*`

Essas permissões permitem que um usuário veja o seguinte:

- Quais usuários, grupos ou funções são anexados a uma política gerenciada
- Quais serviços um usuário ou uma função pode acessar
- A última vez em que acessaram o serviço
- A última vez que tentaram usar uma ação específica do Amazon EC2, IAM, Lambda ou Amazon S3

Para usar a AWS CLI ou a API da AWS para visualizar as informações do último acesso do IAM, você deve ter permissões correspondentes à operação que deseja usar:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

Este exemplo mostra como você pode criar uma política baseada em identidade que permite visualizar as informações do último acesso ao IAM. Além disso, permite o acesso somente leitura a todo o IAM. Esta política define permissões para acesso programático e do console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateServiceLastAccessedDetails",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
```

```
}
```

Permissões para informações do AWS Organizations

Para usar o console do IAM para visualizar um relatório de entidades da raiz, UO ou conta no AWS Organizations, você deve ter uma política que inclua as seguintes ações:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Para usar a API da AWS CLI ou a AWS para visualizar as informações de serviço acessadas por último do AWS Organizations, você deve ter uma política que inclua as seguintes ações:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Este exemplo mostra como você pode criar uma política baseada em identidade que permite visualizar os serviços acessados por último para AWS Organizations. Além disso, permite o acesso

somente leitura a todos do AWS Organizations. Esta política define permissões para acesso programático e do console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

Você também pode usar a chave de condição iam:OrganizationsPolicyId para permitir a geração de um relatório apenas de uma determinada política do AWS Organizations. Para visualizar um exemplo de política, consulte IAM: visualizar serviço acessado por último para uma política do AWS Organizations.

Atividades de solução de problemas para entidades do IAM e do AWS Organizations

Em alguns casos, a tabela de informações acessadas por último do Console de gerenciamento da AWS pode estar vazia. Ou talvez sua solicitação da AWS CLI ou da API da AWS retorne um conjunto vazio de informações ou um campo nulo. Nesses casos, analise os seguintes problemas:

- Para informações acessadas pela última vez pela ação, uma ação que você está esperando ver pode não ser retornada na lista. Isso pode acontecer porque a identidade do IAM não tem permissões para a ação ou a AWS ainda não rastreia a ação quanto a informações do acesso mais recente.
- Para um usuário do IAM, verifique se o usuário tem pelo menos uma política gerenciada ou em linha anexada, diretamente ou por meio de associações de grupo.
- Para um grupo do IAM, verifique se o grupo tem pelo menos uma política em linha ou gerenciada anexada.

- Para um grupo do IAM, o relatório só retorna as informações do último acesso ao serviço de membros que usaram as políticas do grupo para acessar um serviço. Para saber se um membro usou outras políticas, revise as informações acessadas por último desse usuário.
- Para uma função do IAM, verifique se a função tem pelo menos uma política em linha ou gerenciada anexada.
- Para uma entidade do IAM (usuário ou função), revise outros tipos de política que possam afetar as permissões dessa entidade. Entre eles estão políticas baseadas em recurso, listas de controle de acesso, políticas do AWS Organizations, limites de permissões do IAM ou políticas de sessão. Para ter mais informações, consulte Tipos de políticas ou Avaliação de políticas para solicitações em uma única conta.
- Para uma política do IAM, verifique se a política gerenciada especificada está anexada a pelo menos um usuário, grupo com membros ou função.
- Para uma entidade (raiz, UO ou conta) do AWS Organizations, certifique-se de que você esteja conectado usando as credenciais da conta gerencial do AWS Organizations.
- Verifique se as SCPs estão habilitadas na raiz da sua organização.
- As informações de ação acessadas por último só estão disponíveis para as ações listadas em Serviços e ações para os quais a ação do IAM acessou informações pela última vez.

Ao fazer alterações, aguarde pelo menos quatro horas para que a atividade seja exibida no relatório do console do IAM. Se usar a AWS CLI ou a API da AWS, você deverá gerar um novo relatório para visualizar as informações atualizadas.

Onde a AWS rastreia informações acessadas por último

A AWS coleta informações de último acesso das regiões padrão da AWS. Quando a AWS adiciona mais regiões, essas regiões são adicionas à seguinte tabela, incluindo a data em que a AWS começou a rastrear informações em cada região.

- Informações de serviço: o período de rastreamento para os serviços é de, no mínimo, 400 dias, ou menos, caso a sua região tenha começado a rastrear esse recurso nos últimos 400 dias.
- Informações de ações: o período de rastreamento das ações de gerenciamento do Amazon S3 teve início em 12 de abril de 2020. O período de rastreamento para ações de gerenciamento do Amazon EC2, do IAM e do Lambda começou em 7 de abril de 2021. O período de rastreamento para ações de gerenciamento de todos os outros serviços começou em 23 de maio de 2023. Se a data de rastreamento de uma região for posterior a 23 de maio de 2023, as informações acessadas pela última vez pela ação para essa região começarão em uma data posterior.

Nome da região	Região	Data de início do rastreamento
Leste dos EUA (Ohio)	us-east-2	27 de outubro de 2017
Leste dos EUA (Norte da Virgínia)	us-east-1	1 de outubro de 2015
Oeste dos EUA (N. da Califórnia)	us-west-1	1 de outubro de 2015
Oeste dos EUA (Oregon)	us-west-2	1 de outubro de 2015
África (Cidade do Cabo)	af-south-1	22 de abril de 2020
Ásia-Pacífico (Hong Kong)	ap-east-1	24 de abril de 2019
Ásia-Pacífico (Hyderabad)	ap-south-2	22 de novembro de 2022
Ásia-Pacífico (Jacarta)	ap-southeast-3	13 de dezembro de 2021
Ásia-Pacífico (Melbourne)	ap-southeast-4	23 de janeiro de 2023
Ásia-Pacífico (Mumbai)	ap-south-1	27 de junho de 2016
Ásia-Pacífico (Osaka)	ap-northeast-3	11 de fevereiro de 2018
Ásia-Pacífico (Seul)	ap-northeast-2	6 de janeiro de 2016
Ásia-Pacífico (Singapura)	ap-southeast-1	1 de outubro de 2015
Ásia-Pacífico (Sydney)	ap-southeast-2	1 de outubro de 2015
Ásia-Pacífico (Tóquio)	ap-northeast-1	1 de outubro de 2015
Canadá (Central)	ca-central-1	28 de outubro de 2017
Europa (Frankfurt)	eu-central-1	1 de outubro de 2015
Europa (Irlanda)	eu-west-1	1 de outubro de 2015
Europa (Londres)	eu-west-2	28 de outubro de 2017

Nome da região	Região	Data de início do rastreamento
Europa (Milão)	eu-south-1	28 de abril de 2020
Europa (Paris)	eu-west-3	18 de dezembro de 2017
Europa (Espanha)	eu-south-2	15 de novembro de 2022
Europa (Estocolmo)	eu-north-1	12 de dezembro de 2018
Europa (Zurique)	eu-central-2	8 de novembro de 2022
Israel (Tel Aviv)	il-central-1	1º de agosto de 2023
Oriente Médio (Barém)	me-south-1	29 de julho de 2019
Oriente Médio (Emirados Árabes Unidos)	me-central-1	30 de agosto de 2022
América do Sul (São Paulo)	sa-east-1	11 de dezembro de 2015
AWS GovCloud (Leste dos EUA)	us-gov-east-1	1º de julho de 2023
AWS GovCloud (Oeste dos EUA)	us-gov-west-1	1º de julho de 2023

Se uma região não estiver listada na tabela anterior, essa região ainda não fornece informações acessadas por último.

Uma região da AWS é uma coleção de recursos da AWS em uma área geográfica. As regiões são agrupadas em partições. As regiões padrão são aquelas que pertencem à partição `aws`. Para obter mais informações sobre as diferentes partições, consulte Formato de nomes de recursos da Amazon (ARNs) na Referência geral da AWS. Para obter mais informações sobre regiões, consulte Sobre regiões da AWS também na Referência geral da AWS.

Visualizar as últimas informações acessadas do IAM

Você pode visualizar informações acessadas pela última vez para o IAM usando o Console de gerenciamento da AWS, a AWS CLI ou a API da AWS. Veja a lista de serviços e suas ações para

as quais as últimas informações acessadas são exibidas. Para obter mais informações sobre as informações acessadas pela última vez, consulte Refinar permissões na AWS usando informações do último acesso.

É possível visualizar as informações para os tipos de recurso a seguir no IAM. Em cada caso, as informações abrangem serviços permitidos para o período de geração de relatórios indicado:

- Usuário: visualize a última vez em que o usuário tentou acessar cada serviço permitido.
- Grupo de usuários: visualize informações sobre a última vez em que um membro do grupo de usuários tentou acessar cada serviço permitido. Esse relatório também inclui o número total de membros que tentou acessar.
- Função: visualize a última vez em que alguém usou a função em uma tentativa de acessar cada serviço permitido.
- Política: visualize informações sobre a última vez em que um usuário ou uma função tentou acessar cada serviço permitido. Esse relatório também inclui o número total de entidades que tentou acessar.

> **Note**
>
> Para visualizar as informações de acesso de um recurso no IAM, compreenda o período de geração de relatórios, as entidades relatadas e os tipos de política avaliados para as informações. Para obter mais detalhes, consulte the section called “Coisas para saber sobre as informações acessadas por último”.

Visualização de informações do IAM (console)

Você pode visualizar as informações do último acesso ao IAM na guia Último acesso no console do IAM.

Para visualizar informações do IAM (console)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha User groups (Grupos de usuários), Users (USuários), Roles (Funções) ou Policies (Políticas).

3. Escolha qualquer nome de usuário, grupo de usuários, perfil ou política para abrir a página Resumo correspondente e escolha a guia Acessados recentemente. Exiba as seguintes informações com base no recurso escolhido:

   - User group (Grupo de usuários): exibe a lista de serviços que os membros do grupo de usuários podem acessar. Você também pode visualizar quando um membro acessou o serviço pela última vez, quais políticas de grupo de usuário ele usou e qual membro do grupo de usuários fez a solicitação. Escolha o nome da política para saber se ela é uma política gerenciada ou uma política de grupo de usuários em linha. Escolha o nome do membro do grupo de usuários para ver todos os membros do grupo de usuários e quando eles acessaram o serviço mais recentemente.
   - User (Usuário): exibe a lista de serviços que o usuário pode acessar. Também é possível exibir quando ele acessou o serviço pela última vez e quais políticas estão associadas ao usuário no momento. Escolha o nome da política para saber se é uma política gerenciada, uma política de usuário em linha ou uma política em linha para o grupo.
   - Role (Função): exibe a lista de serviços que a função pode acessar, quando a função acessou o serviço mais recentemente e quais políticas foram usadas. Escolha o nome da política para saber se ela é uma política gerenciada ou uma política de função em linha.
   - Policy (Política): exibe a lista de serviços com ações permitidas na política. Você também pode exibir quando a política foi usada pela última vez para acessar o serviço e qual entidade (usuário ou função) usou a política. A data de Último acesso também inclui quando o acesso é concedido a essa política por meio de outra política. Escolha o nome da entidade para saber quais entidades têm essa política anexada e quando elas acessaram o serviço mais recentemente.

4. Na coluna Serviço da tabela, escolha o nome de um dos serviços que inclui informações acessadas pela última vez pela ação para visualizar uma lista de ações de gerenciamento que as entidades do IAM tentaram acessar. É possível visualizar a Região da AWS e um carimbo de data/hora que mostre quando alguém tentou executar a ação pela última vez.

5. A coluna Último acesso é exibida para serviços e ações de gerenciamento dos serviços que incluem informações acessadas pela última vez pela ação. Revise os seguintes resultados possíveis que são retornados nessa coluna. Esses resultados variam dependendo se um serviço ou ação é permitido, foi acessado e se ele é rastreado pela AWS quanto a informações acessadas por último.

<number of> dias atrás

O número de dias desde que o serviço ou a ação foi usado no período de rastreamento. O período de rastreamento dos serviços é para os últimos 400 dias. O período de rastreamento das ações do Amazon S3 começou em 12 de abril de 2020. O período de rastreamento das ações do Amazon EC2, do IAM e do Lambda começou em 7 de abril de 2021. O período de rastreamento para todos os outros serviços começou em 23 de maio de 2023. Para saber mais sobre as datas de início de rastreamento para cada Região da AWS, consulte Onde a AWS rastreia informações acessadas por último.

Não acessado no período de rastreamento

O serviço ou ação rastreado não foi usado por uma entidade no período de rastreamento.

É possível que você tenha permissões para uma ação que não aparece na lista. Isso poderá acontecer se as informações de rastreamento da ação não tiverem sido incluídas pela AWS. Você não deve tomar decisões de permissões com base apenas na ausência de informações de rastreamento. Em vez disso, recomendamos que você use essas informações para informar e apoiar sua estratégia geral de concessão de privilégios mínimos. Verifique suas políticas para confirmar se o nível de acesso é apropriado.

Visualização de informações do IAM (AWS CLI)

Você pode usar a AWS CLI para recuperar informações sobre a última vez que um recurso do IAM foi usado para tentar acessar produtos da AWS e ações do Amazon S3, do Amazon EC2, do IAM e do Lambda. Um recurso do IAM pode ser um usuário, grupo de usuários, função ou política.

Para visualizar informações do IAM (AWS CLI)

1. Gere um relatório. A solicitação deve incluir o ARN do recurso do IAM (usuário, grupo de usuários, função ou política) para o qual você deseja um relatório. Você pode especificar o nível de granularidade que deseja gerar no relatório para exibir detalhes de acesso para serviços ou serviços e ações. A solicitação retorna um `job-id` que você pode acabar usando nas operações `get-service-last-accessed-details` e `get-service-last-accessed-details-with-entities` para monitorar o `job-status` até o trabalho estar concluído.

   - aws iam generate-service-last-accessed-details

2. Recupere detalhes sobre o relatório usando o parâmetro `job-id` da etapa anterior.

- aws iam get-service-last-accessed-details

Essa operação retorna as seguintes informações, com base no tipo de recurso e nível de granularidade que você solicitou na operação `generate-service-last-accessed-details`:

- Usuário: retorna uma lista de serviços que o usuário especificado pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa do usuário e o ARN do usuário.
- Grupo de usuários:retorna uma lista de serviços que os membros do grupo de usuários especificado podem acessar usando políticas anexadas ao grupo de usuários. Para cada serviço, a operação retorna a data e a hora da última tentativa feita por qualquer membro do grupo de usuários. Ela também retorna o ARN do usuário e o número total de membros do grupode usuários que tentou acessar o serviço. Use a operação GetServiceLastAccessedDetailsWithEntities para recuperar uma lista de todos os membros.
- Função: retorna uma lista de serviços que a função especificada pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa da função e o ARN da função.
- Política: retorna uma lista de serviços para os quais a política especificada permite o acesso. Para cada serviço, a operação retorna a data e a hora em que uma entidade (usuário ou função) tentou acessar mais recentemente o serviço usando a política. Ele também retorna o ARN dessa entidade e o número total de entidades que tentou acessar.

3. Saiba mais sobre as entidades que usaram permissões de grupo de usuários ou política em uma tentativa de acessar um serviço específico. Essa operação retorna uma lista de entidades com ARN, ID, nome, caminho, tipo (usuário ou função) de cada entidade e quando eles tentaram acessar o serviço mais recentemente. Você também pode usar essa operação para usuários e funções, mas ela só retorna informações sobre essa entidade.

   - aws iam get-service-last-accessed-details-with-entities

4. Saiba mais sobre as políticas baseadas em identidade que uma identidade (usuário, grupo de usuários ou função) usou em uma tentativa de acessar um serviço específico. Quando você especifica uma identidade e um serviço, essa operação retorna uma lista de políticas de permissões que a identidade pode usar para acessar o serviço especificado. Essa operação indica o estado atual de políticas e não depende do relatório gerado. Isso também não retorna outros tipos de política, como políticas baseadas em recurso, listas de controle de acesso, políticas do AWS Organizations, limites de permissões do IAM ou políticas de sessão. Para ter

mais informações, consulte Tipos de políticas ou Avaliação de políticas para solicitações em uma única conta.

- aws iam list-policies-granting-service-access

Visualização de informações do IAM (API da AWS)

Você pode usar a API da AWS para recuperar informações sobre a última vez que um recurso do IAM foi usado para tentar acessar produtos da AWS e ações do Amazon S3, do Amazon EC2, do IAM e do Lambda. Um recurso do IAM pode ser um usuário, grupo de usuários, função ou política. Você pode especificar o nível de granularidade a ser gerado no relatório para exibir detalhes de serviços ou serviços e ações.

Para visualizar informações do IAM (API da AWS)

1. Gere um relatório. A solicitação deve incluir o ARN do recurso do IAM (usuário, grupo de usuários, função ou política) para o qual você deseja um relatório. Ele retorna um `JobId` que você pode acabar usando nas operações `GetServiceLastAccessedDetails` e `GetServiceLastAccessedDetailsWithEntities` para monitorar o `JobStatus` até o trabalho estar concluído.

   - GenerateServiceLastAccessedDetails
2. Recupere detalhes sobre o relatório usando o parâmetro `JobId` da etapa anterior.

   - GetServiceLastAccessedDetails

   Essa operação retorna as seguintes informações, com base no tipo de recurso e nível de granularidade que você solicitou na operação `GenerateServiceLastAccessedDetails`:

   - Usuário: retorna uma lista de serviços que o usuário especificado pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa do usuário e o ARN do usuário.
   - Grupo de usuários:retorna uma lista de serviços que os membros do grupo de usuários especificado podem acessar usando políticas anexadas ao grupo de usuários. Para cada serviço, a operação retorna a data e a hora da última tentativa feita por qualquer membro do grupo de usuários. Ela também retorna o ARN do usuário e o número total de membros do grupode usuários que tentou acessar o serviço. Use a operação GetServiceLastAccessedDetailsWithEntities para recuperar uma lista de todos os membros.

- Função: retorna uma lista de serviços que a função especificada pode acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa da função e o ARN da função.
- Política: retorna uma lista de serviços para os quais a política especificada permite o acesso. Para cada serviço, a operação retorna a data e a hora em que uma entidade (usuário ou função) tentou acessar mais recentemente o serviço usando a política. Ele também retorna o ARN dessa entidade e o número total de entidades que tentou acessar.

3. Saiba mais sobre as entidades que usaram permissões de grupo de usuários ou política em uma tentativa de acessar um serviço específico. Essa operação retorna uma lista de entidades com ARN, ID, nome, caminho, tipo (usuário ou função) de cada entidade e quando eles tentaram acessar o serviço mais recentemente. Você também pode usar essa operação para usuários e funções, mas ela só retorna informações sobre essa entidade.

   - GetServiceLastAccessedDetailsWithEntities

4. Saiba mais sobre as políticas baseadas em identidade que uma identidade (usuário, grupo de usuários ou função) usou em uma tentativa de acessar um serviço específico. Quando você especifica uma identidade e um serviço, essa operação retorna uma lista de políticas de permissões que a identidade pode usar para acessar o serviço especificado. Essa operação indica o estado atual de políticas e não depende do relatório gerado. Isso também não retorna outros tipos de política, como políticas baseadas em recurso, listas de controle de acesso, políticas do AWS Organizations, limites de permissões do IAM ou políticas de sessão. Para ter mais informações, consulte Tipos de políticas ou Avaliação de políticas para solicitações em uma única conta.

   - ListPoliciesGrantingServiceAccess

Visualizar as últimas informações acessadas do AWS Organizations

Você pode visualizar as informações do último acesso ao serviço do AWS Organizations usando o console do IAM, a AWS CLI ou a API da AWS. Para obter informações importantes sobre os dados, permissões necessárias, solução de problemas e regiões com suporte, consulte Refinar permissões na AWS usando informações do último acesso.

Ao fazer login no console do IAM usando as credenciais da conta gerencial do AWS Organizations, você pode visualizar as informações de qualquer entidade na sua organização. As entidades do AWS Organizations incluem a raiz da organização, unidades organizacionais (UOs) e contas. Você também pode usar o console do IAM para visualizar informações sobre quaisquer políticas de controle de serviço (SCPs) na sua organização. O IAM mostra uma lista de serviços que são

permitidos por quaisquer SCPs que se aplicam à entidade. Para cada serviço, você pode visualizar as informações mais recentes de atividades da conta para a entidade do AWS Organizations escolhida ou para os filhos da entidade.

Ao usar a AWS CLI ou a API da AWS com credenciais de gerenciamento da conta, você pode gerar um relatório para quaisquer entidades ou políticas em sua organização. Um relatório programático para uma entidade inclui uma lista de serviços que são permitidos pelas SCPs que se aplicam à entidade. Para cada serviço, o relatório inclui a atividade mais recente de contas na entidade do AWS Organizations especificada ou a subárvore da entidade.

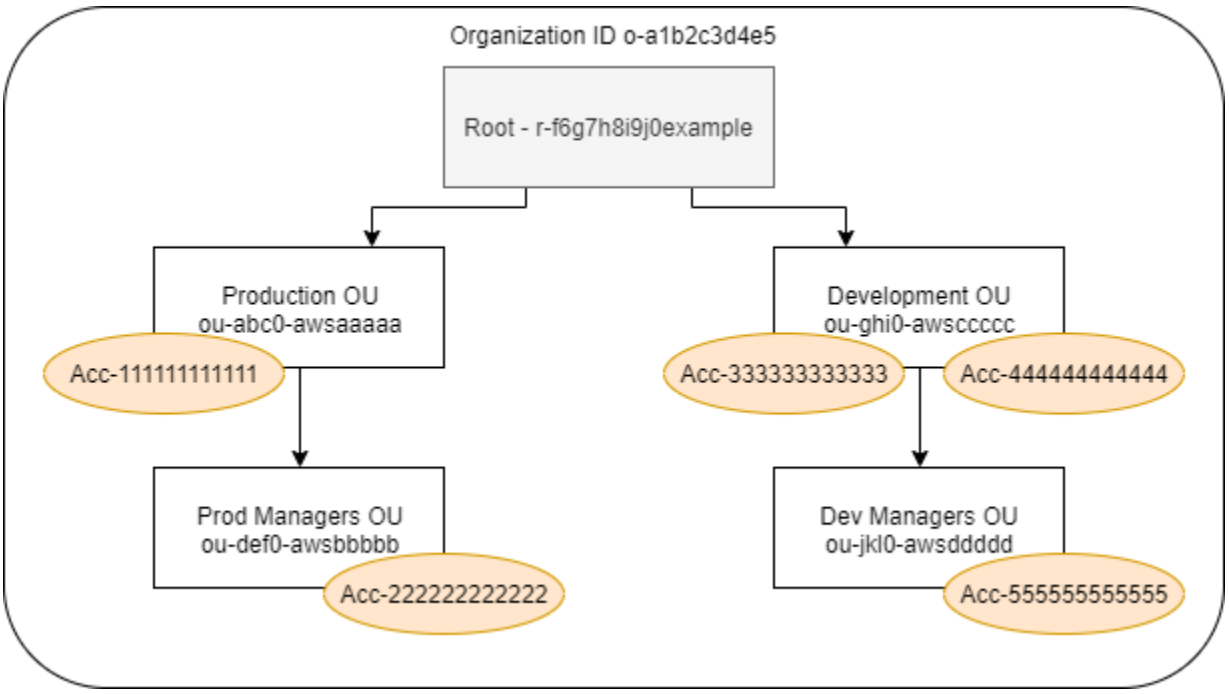
Ao gerar um relatório programático para uma política, você deve especificar uma entidade do AWS Organizations. Esse relatório inclui uma lista de serviços que são permitidos pela SCP especificada. Para cada serviço, ele inclui a atividade de conta mais recente na entidade ou filhos da entidade filhos que recebem permissão por essa política. Para obter mais informações, consulte aws iam generate-organizations-access-report ou GenerateOrganizationsAccessReport.

Antes de visualizar o relatório, é necessário compreender as informações e os requisitos da conta de gerenciamento, o período relatado, as entidades relatadas e os tipos de política avaliados. Para obter mais detalhes, consulte the section called “Coisas para saber sobre as informações acessadas por último”.

Compreender o caminho da entidade do AWS Organizations

Ao usar a AWS CLI ou a API da AWS para gerar um relatório de acesso do AWS Organizations, você deverá especificar um caminho de entidade. Um caminho é uma representação de texto da estrutura de uma entidade do AWS Organizations.

Você pode criar um caminho de entidade usando a estrutura conhecida de sua organização. Por exemplo, suponha que você tenha a estrutura organizacional a seguir no AWS Organizations.



O caminho para a UO Dev Managers (Gerentes de Desenvolvimento) é criado usando os IDs da organização, da raiz e de todas as UOs no caminho até, e incluindo, a UO.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsdddddd/
```

O caminho para a conta na UO Production (Produção) é criado usando os IDs da organização, da raiz, da UO e o número da conta.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> Os IDs de organização são globalmente exclusivos, mas os IDs da UO e da raiz são exclusivos somente dentro de uma organização. Isso significa que duas organizações não compartilham o mesmo ID de organização. No entanto, outra organização pode ter uma UO ou raiz com o mesmo ID que o seu. Recomendamos sempre incluir o ID da organização ao especificar uma UO ou raiz.

Visualizar informações do AWS Organizations (console)

Você pode usar o console do IAM para visualizar informações do último acesso ao serviço para sua raiz, UO, conta ou política.

Como visualizar informações para a raiz (console)

1. Faça login no Console de gerenciamento da AWS usando as credenciais da conta gerencial do AWS Organizations e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação abaixo da seção Access reports (Relatórios de acesso), selecione Organization activity (Atividade da organização).
3. Na página Organization activity (Atividade da organização), selecione Root (Raiz).
4. Na guia Details and activity (Atividade e detalhes), visualize a seção Service access report (Relatório de acesso ao serviço). As informações incluem uma lista de serviços que são permitidos pelas políticas anexadas diretamente na raiz. As informações mostram de qual conta o serviço foi acessado mais recentemente e quando. Para obter mais detalhes sobre qual entidade de segurança acessou o serviço, faça login como administrador nessa conta e visualize as informações de serviço acessadas por último do IAM.
5. Escolha a guia Attached SCPs (SCPs anexadas) para visualizar a lista de políticas de controle de serviço (SCPs) anexadas à raiz. O IAM mostra o número de entidades de destino às quais cada política está anexada. Você pode usar essas informações para decidir qual SCP analisar.
6. Escolha o nome de uma SCP para visualizar todos os serviços que a política permite. Para cada serviço, visualize de qual conta o serviço foi acessado mais recentemente e quando.
7. Selecione Editar no AWS Organizations para visualizar detalhes adicionais e editar a SCP no console do AWS Organizations. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS Organizations.

Como visualizar informações de uma UO ou conta (console)

1. Faça login no Console de gerenciamento da AWS usando as credenciais da conta gerencial do AWS Organizations e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação abaixo da seção Access reports (Relatórios de acesso), selecione Organization activity (Atividade da organização).
3. Na página Organization activity (Atividade da organização), expanda a estrutura da sua organização. Em seguida, escolha o nome da UO ou qualquer conta que você queira visualizar, exceto a conta de gerenciamento.

4. Na guia Details and activity (Atividade e detalhes), visualize a seção Service access report (Relatório de acesso ao serviço). As informações incluem uma lista de serviços que são permitidos pelas SCPs anexadas à UO ou conta e todos os seus pais. As informações mostram de qual conta o serviço foi acessado mais recentemente e quando. Para obter mais detalhes sobre qual entidade de segurança acessou o serviço, faça login como administrador nessa conta e visualize as informações de serviço acessadas por último do IAM.
5. Escolha a guia Attached SCPs (SCPs anexadas) para visualizar a lista de políticas de controle de serviço (SCPs) que estão diretamente anexadas à UO ou à conta. O IAM mostra o número de entidades de destino às quais cada política está anexada. Você pode usar essas informações para decidir qual SCP analisar.
6. Escolha o nome de uma SCP para visualizar todos os serviços que a política permite. Para cada serviço, visualize de qual conta o serviço foi acessado mais recentemente e quando.
7. Selecione Editar no AWS Organizations para visualizar detalhes adicionais e editar a SCP no console do AWS Organizations. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS Organizations.

Para visualizar as informações da conta de gerenciamento (console)

1. Faça login no Console de gerenciamento da AWS usando as credenciais da conta gerencial do AWS Organizations e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação abaixo da seção Access reports (Relatórios de acesso), selecione Organization activity (Atividade da organização).
3. Na página Organization activity (Atividade da organização), expanda a estrutura da sua organização e escolha o nome da sua conta de gerenciamento.
4. Na guia Details and activity (Atividade e detalhes), visualize a seção Service access report (Relatório de acesso ao serviço). As informações incluem uma lista de todos os serviços da AWS. A conta de gerenciamento não é limitada pelas SCPs. As informações mostram se a conta acessou o serviço mais recentemente e quando. Para obter mais detalhes sobre qual entidade de segurança acessou o serviço, faça login como administrador nessa conta e visualize as informações de serviço acessadas por último do IAM.
5. Escolha Attached SCPs (SCPs anexadas) para confirmar que não há SCPs anexadas, pois a conta é a conta de gerenciamento.

Como visualizar informações de uma política (console)

1. Faça login no Console de gerenciamento da AWS usando as credenciais da conta gerencial do AWS Organizations e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação abaixo da seção Access reports (Relatórios de acesso), selecione Service control policies (SCPs) (Políticas de controle de serviço (SCPs)).
3. Na página Políticas de controle de serviço (SCPs), visualize uma lista de políticas da sua organização. Você pode visualizar o número de entidades de destino às quais cada política está anexada.
4. Escolha o nome de uma SCP para visualizar todos os serviços que a política permite. Para cada serviço, visualize de qual conta o serviço foi acessado mais recentemente e quando.
5. Selecione Editar no AWS Organizations para visualizar detalhes adicionais e editar a SCP no console do AWS Organizations. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS Organizations.

Visualizar informações do AWS Organizations (AWS CLI)

Você pode usar a AWS CLI para recuperar informações de serviço acessadas por último de sua raiz do AWS Organizations, UO ou política.

Como visualizar informações de serviço acessadas por último do AWS Organizations (AWS CLI)

1. Use as credenciais da conta gerencial do AWS Organizations com as permissões necessárias do IAM e do AWS Organizations e confirme se as SCPs estão habilitadas para a raiz. Para obter mais informações, consulte Coisas para saber sobre as informações acessadas por último.
2. Gere um relatório. A solicitação deve incluir o caminho da entidade do AWS Organizations (raiz, UO ou conta) para a qual você deseja um relatório. Você também pode incluir um parâmetro `organization-policy-id` para visualizar um relatório para uma política específica. O comando retorna um `job-id` que você pode usar no comando `get-organizations-access-report` e para monitorar o `job-status` até o trabalho ser concluído.
   - aws iam generate-organizations-access-report
3. Recupere detalhes sobre o relatório usando o parâmetro `job-id` da etapa anterior.
   - aws iam get-organizations-access-report

Esse comando retorna uma lista de serviços que membros de entidade podem acessar. Para cada serviço, o comando retorna a data e a hora de uma conta membro da última tentativa e o caminho de entidade da conta. Isso também retorna o número total de serviços que estão disponíveis para acesso e o número de serviços que não foram acessados. Se você tiver especificado o parâmetro opcional `organizations-policy-id`, os serviços que estão disponíveis para acesso são aqueles que são permitidos pela política especificada.

Visualizar informações do AWS Organizations (API da AWS)

Você pode usar a API da AWS para recuperar informações de serviço acessadas por último para sua raiz do AWS Organizations, UO, conta ou política.

Como visualizar informações de serviço acessadas por último do AWS Organizations (API da AWS)

1. Use as credenciais da conta gerencial do AWS Organizations com as permissões necessárias do IAM e do AWS Organizations e confirme se as SCPs estão habilitadas para a raiz. Para obter mais informações, consulte Coisas para saber sobre as informações acessadas por último.
2. Gere um relatório. A solicitação deve incluir o caminho da entidade do AWS Organizations (raiz, UO ou conta) para a qual você deseja um relatório. Você também pode incluir um parâmetro `OrganizationsPolicyId` para visualizar um relatório para uma política específica. A operação retorna um `JobId` que você pode usar na operação `GetOrganizationsAccessReport` para monitorar o `JobStatus` até o trabalho ser concluído.

   - GenerateOrganizationsAccessReport
3. Recupere detalhes sobre o relatório usando o parâmetro `JobId` da etapa anterior.

   - GetOrganizationsAccessReport

   Essa operação retorna uma lista de serviços que membros de entidade podem acessar. Para cada serviço, a operação retorna a data e a hora da última tentativa de um membro da conta e o caminho de entidade da conta. Isso também retorna o número total de serviços que estão disponíveis para acesso e o número de serviços que não foram acessados. Se você tiver especificado o parâmetro opcional `OrganizationsPolicyId`, os serviços que estão disponíveis para acesso são aqueles que são permitidos pela política especificada.

Cenários de exemplo para uso de informações acessadas por último

Você pode usar as informações do último acesso para tomar decisões sobre as permissões concedidas às entidades do IAM ou do AWS Organizations. Para obter mais informações, consulte Refinar permissões na AWS usando informações do último acesso.

> **Note**
>
> Antes de visualizar as informações de acesso para uma entidade ou política no IAM ou no AWS Organizations, é necessário entender o período do relatório, as entidades relatadas e os tipos de política avaliados para seus dados. Para obter mais detalhes, consulte the section called “Coisas para saber sobre as informações acessadas por último”.

Cabe a você, como administrador, equilibrar a acessibilidade e o privilégio mínimo apropriado para a organização.

Uso de informações para reduzir permissões para um grupo do IAM

Você pode usar as informações do último acesso para reduzir permissões do grupo do IAM para incluir apenas os serviços de que os usuários precisam. Esse método é uma etapa importante em conceder menor privilégio em um nível de serviço.

Por exemplo, Paulo Santos é o administrador responsável por definir as permissões de usuário da AWS para Exemplo Corp. Esta empresa acabou de começar a usar a AWS, e a equipe de desenvolvimento de software ainda não definiu quais produtos da AWS serão usados. Paulo deseja dar à equipe permissão para acessar apenas os serviços de que precisam, mas como ainda não está definida, ele dá temporariamente a eles permissões de superusuário. Depois, ele usa as informações acessadas por último para reduzir as permissões do grupo.

Paulo cria uma política gerenciada chamada `ExampleDevelopment` usando o texto JSON a seguir. Em seguida, ele o anexa a um grupo chamado `Development` e adiciona todos os desenvolvedores ao grupo.

> **Note**
>
> Os usuários avançados do Paulo podem precisar de `iam:CreateServiceLinkedRole` permissões para usar alguns serviços e recursos. Ele entende que a adição dessa permissão

permite que os usuários criem qualquer função vinculada a serviços. Ele aceita esse risco para seus usuários avançados.

JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo opta por aguardar 90 dias até visualizar as informações acessadas por último para o grupo `Development` usando o Console de gerenciamento da AWS. Ele exibe a lista de serviços que os membros do grupo acessaram. Ele aprende que os usuários acessaram cinco serviços na última semana: AWS CloudTrail, Amazon CloudWatch Logs, Amazon EC2, AWS KMS e Amazon S3. Eles acessaram alguns outros serviços quando estavam avaliando inicialmente a AWS, mas não desde então.

Paulo opta por reduzir as permissões de política para só incluir os cinco serviços e as ações do IAM e do AWS Organizations obrigatórios. Ele edita a política `ExampleDevelopment` usando o texto JSON a seguir.

> **Note**
>
> Os usuários avançados do Paulo podem precisar de `iam:CreateServiceLinkedRole` permissões para usar alguns serviços e recursos. Ele entende que a adição dessa permissão permite que os usuários criem qualquer função vinculada a serviços. Ele aceita esse risco para seus usuários avançados.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Para reduzir ainda mais as permissões, Paulo pode exibir os eventos da conta no Event history (Histórico de eventos) da AWS CloudTrail. Lá ele pode exibir informações detalhadas que pode usar para reduzir as permissões da política a fim de incluir apenas as ações e os recursos de que os desenvolvedores precisam. Para obter mais informações, consulte Visualizar eventos do CloudTrail no console do CloudTrail no Guia do usuário do AWS CloudTrail.

Uso de informações para reduzir permissões para um usuário do IAM

Você pode usar informações do último acesso para reduzir as permissões de um usuário do IAM individual.

Por exemplo, Martha Rivera é a administradora de TI responsável por garantir que as pessoas na organização não tenham permissões da AWS em excesso. Como parte de uma verificação de segurança periódica, ela revisa as permissões de todos os usuários do IAM. Um desses usuários é um desenvolvedor de aplicativos chamado Nikhil Jayashankar que, anteriormente, assumiu a função de um engenheiro de segurança. Por causa da alteração feita em requisitos de trabalho, Nikhil é membro dos grupos `app-dev` e `security-team`. O grupo `app-dev` de seu novo trabalho concede permissões para vários serviços, incluindo Amazon EC2, Amazon EBS, Auto Scaling, Amazon S3, Route 53 e Elastic Transcoder. O grupo `security-team` de seu trabalho anterior concede permissões ao IAM e ao CloudTrail.

Como administradora, Martha inicia uma sessão no console do IAM e escolhe Usuários, o nome `nikhilj` e depois a guia Acessados recentemente.

Martha revisa a coluna Acessados recentemente e nota que Nikhil não acessou recentemente o IAM, o CloudTrail, o Route 53, o Amazon Elastic Transcoder e vários outros serviços da AWS. Nikhil acessou o Amazon S3. Martha escolhe S3 na lista de serviços e vê que Nikhil realizou algumas ações `List` do Amazon S3 nas últimas duas semanas. Dentro da empresa, Martha confirma que Nikhil não precisa mais acessar o IAM e o CloudTrail, porque ele não é mais membro da equipe de segurança interna.

Martha agora está pronta para agir sobre as informações de serviço e ação acessadas por último. No entanto, diferentemente do grupo no exemplo anterior, um usuário do IAM como `nikhilj` pode estar sujeito a várias políticas e ser membro de vários grupos. Martha deve prosseguir com cuidado para não interromper inadvertidamente o acesso para `nikhilj` ou outros membros do grupo. Além de aprender o acesso que Nikhil deve ter, ela deve determinar como ele está recebendo essas permissões.

Martha escolhe a guia Permissions (Permissões), em que ela exibe quais políticas estão anexadas diretamente a `nikhilj` e as anexadas de um grupo. Ela expande cada política e exibe o resumo da política para saber qual política dá acesso a serviços que Nikhil não está usando:

- IAM: a política gerenciada pela AWS `IAMFullAccess` é anexada diretamente a `nikhilj` e anexada ao grupo `security-team`.
- CloudTrail: a política gerenciada pela AWS `AWSCloudTrailReadOnlyAccess` é anexada ao grupo `security-team`.
- Route 53: a política gerenciada pelo cliente `App-Dev-Route53` é anexada ao grupo `app-dev`.
- Elastic Transcoder: a política gerenciada pelo cliente `App-Dev-ElasticTranscoder` é anexada ao grupo `app-dev`.

Martha opta por remover a política gerenciada pela `IAMFullAccess` AWS anexada diretamente a `nikhilj`. Ela também remove a participação de Nikhil no grupo `security-team`. Essas duas ações removem o acesso desnecessário ao IAM e ao CloudTrail.

As permissões de Nikhil para acessar o Route 53 e o Elastic Transcoder são concedidas pelo grupo `app-dev`. Embora Nikhil não esteja usando esses serviços, outros membros do grupo podem estar. Martha analisa as informações do último acesso para o grupo `app-dev` e descobre que vários membros acessaram recentemente o Route 53 e o Amazon S3. Mas nenhum membro do grupo acessou o Elastic Transcoder no último ano. Ela remove a política gerenciada pelo cliente `App-Dev-ElasticTranscoder` do grupo.

Martha acaba revisando as informações acessadas por último para a política gerenciada pelo cliente `App-Dev-ElasticTranscoder`. Ela descobre que a política não está anexada a nenhuma outra identidade do IAM. Ela investiga dentro da empresa para garantir que a política não será necessária no futuro e, em seguida, ela a excluirá.

Uso das informações antes de excluir recursos do IAM

Você pode usar informações do último acesso antes de excluir um recurso do IAM para garantir que um determinado período tenha se passado desde a última vez em que alguém usou o recurso. Isso se aplica a usuários, grupos, funções e políticas. Para saber mais sobre essas ações, consulte os seguintes tópicos:

- Usuários do IAM: remoção ou desativação de um usuário do IAM
- Grupos: exclusão de um grupo do IAM

- Funções: excluir perfis ou perfis de instância
- Políticas: excluir políticas do IAM (essa ação também desvincula a política das identidades)

Uso de informações antes de editar políticas do IAM

Você pode revisar as informações do último acesso de uma identidade (usuário, grupo ou função) do IAM ou de uma política do IAM antes de editar uma política que afete esse recurso. Isso é importante porque você não deseja remover acesso para alguém que a esteja usando.

Por exemplo, Arnav Desai é desenvolvedor e administrador da AWS da Exemplo Corp. Quando sua equipe começou a usar a AWS, foi concedido a todos os desenvolvedores acesso de usuário avançado, o que permitiu a eles ter acesso total a todos os serviços, exceto o IAM e o AWS Organizations. Como uma primeira etapa para conceder o menor privilégio, Arnav deseja usar a AWS CLI para revisar políticas gerenciadas na conta.

Para isso, Arnav primeiro lista as políticas de permissões gerenciadas pelo cliente na conta anexadas a uma identidade usando o seguinte comando:

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

Na resposta, ele captura o ARN de cada política. Depois disso, Arnav gera um relatório de informações acessadas por último para cada política usando o comando a seguir.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Nessa resposta, ele captura o ID do relatório gerado com base no campo `JobId`. Em seguida, Arnav sondará o comando a seguir até o campo `JobStatus` retornar um valor de `COMPLETED` ou `FAILED`. Se a tarefa falhar, ele vai capturar o erro.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Quando o trabalho tem um status `COMPLETED`, Arnav analisa o conteúdo da matriz `ServicesLastAccessed` formatada em JSON.

```
 "ServicesLastAccessed": [
       {
           "TotalAuthenticatedEntities": 1,
```

```
            "LastAuthenticated": 2018-11-01T21:24:33.222Z,
            "ServiceNamespace": "dynamodb",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
            "ServiceName": "Amazon DynamoDB"
        },

        {
            "TotalAuthenticatedEntities": 0,
            "ServiceNamespace": "ec2",
            "ServiceName": "Amazon EC2"
        },

        {
            "TotalAuthenticatedEntities": 3,
            "LastAuthenticated": 2018-08-25T15:29:51.156Z,
            "ServiceNamespace": "s3",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
            "ServiceName": "Amazon S3"
        }
    ]
```

Com base nessas informações, Arnav descobre que a política `ExamplePolicy1` permite acesso a três serviços, Amazon DynamoDB, Amazon S3 e Amazon EC2. O usuário do IAM chamado `IAMExampleUser` tentou acessar o DynamoDB pela última vez em 1.º de novembro e alguém usou a função `IAMExampleRole` para tentar acessar o Amazon S3 em 25 de agosto. Também existem mais duas entidades que tentaram acessar o Amazon S3 no último ano. No entanto, ninguém tentou acessar o Amazon EC2 no último ano.

Isso significa que o Arnav pode remover com segurança as ações do Amazon EC2 da política. Arnav deseja revisar o documento JSON atual da política. Primeiro, ele deve determinar o número da versão da política usando o comando a seguir.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Na resposta, Arnav coleta o número da versão padrão atual da matriz `Versions`. Ele acaba usando esse número de versão (`v2`) para solicitar o documento de política JSON usando o comando a seguir.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav armazena o documento de política JSON retornado no campo `Document` da matriz `PolicyVersion`. No documento de política, Arnav procura ações com o namespace `ec2`. Se não houver ações de outros namespaces restantes na política, ele desanexará a política das identidades afetadas (usuários, grupos e funções). Depois disso, ele exclui a política. Nesse caso, a política inclui os serviços Amazon DynamoDB e Amazon S3. Portanto, Arnav remove as ações do Amazon EC2 do documento e salva suas alterações. Ele acaba usando o comando a seguir para atualizar a política usando a nova versão do documento e definir essa versão como a versão da política padrão.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

A política `ExamplePolicy1` agora é atualizada para remover o acesso ao serviço do Amazon EC2 desnecessário.

Outros cenários do IAM

As informações sobre quando um recurso (usuário, grupo, função ou política) do IAM tentou acessar pela última vez um serviço podem ajudar quando você concluir qualquer uma das seguintes tarefas:

- Políticas: Editar uma política em linha ou gerenciada pelo cliente existente para remover permissões
- Políticas: Converter uma política em linha em uma política gerenciada e excluí-la
- Políticas: Adicionar uma negação explícita a uma política existente
- Políticas: Desvincular uma política gerenciada de uma identidade (usuário, grupo ou função)
- Entidades: Definir um limite de permissões para controlar as permissões máximas que uma entidade (usuário ou função) pode ter
- Grupos: Remover usuários de um grupo

Usar informações para refinar permissões para uma unidade organizacional

Você pode usar informações acessadas por último para refinar as permissões para uma unidade organizacional (UO) no AWS Organizations.

Por exemplo, John Stiles é um administrador do AWS Organizations. Ele é responsável por garantir que as pessoas nas Contas da AWS da empresa não tenham permissões em excesso. Como parte de uma auditoria de segurança periódica, ele analisa as permissões da sua organização. A UO `Development` contém contas que são frequentemente usadas para testar novos serviços da AWS.

John decide analisar periodicamente o relatório de serviços que não foram acessados em mais de 180 dias. Depois disso, ele remove as permissões de acesso dos membros da UO a esses serviços.

John faz login no console do IAM usando as credenciais de sua conta de gerenciamento. No console do IAN, ele localiza os dados do AWS Organizations para a UO `Development`. Ele analisa a tabela Service access report (Relatório de acesso ao serviço) e vê dois serviços da AWS que não foram acessados em mais de 180 dias (o período preferencial). Ele lembra-se de adicionar permissões para que as equipes de desenvolvimento acessem o Amazon Lex e o AWS Database Migration Service. John entra em contato com as equipes de desenvolvimento e confirma que eles não têm mais a necessidade de testar esses serviços.

John agora está pronto para agir com base nas informações acessadas por último. Ele escolhe Editar no AWS Organizations e lembra-se de que a SCP está anexada a várias entidades. Ele escolhe Continue (Continuar). No AWS Organizations, ele analisa os destinos para saber a quais entidades do AWS Organizations a SCP está anexada. Todas as entidades estão na UO `Development`.

John decide negar acesso às ações do Amazon Lex e do AWS Database Migration Service na SCP `NewServiceTest`. Essa ação remove o acesso desnecessário aos serviços.

Serviços e ações para os quais a ação do IAM acessou informações pela última vez

A tabela a seguir lista os serviços da AWS para os quais a ação do IAM acessou informações pela última vez. Para obter uma lista de ações em cada serviço, consulte Ações, recursos e chaves de condição dos serviços da AWS na Referência de autorização do serviço.

AWSA fornece informações de ação de último acesso no formato JSON para agilizar a automação dos fluxos de trabalho de gerenciamento de políticas. Com as informações de referência do serviço, você pode acessar as informações de ação do último acesso em Serviços da AWS de arquivos legíveis por máquina. Para obter mais informações, consulte Simplified AWS service (Serviço da AWS) information for programmatic access na Service Authorization Reference.

Serviço	Prefixo do serviço
AWS Identity and Access Management and Access Analyzer	access-analyzer
AWS Gerenciamento de contas	conta
Gerenciador de certificados da AWS	acm

Serviço	Prefixo do serviço
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Application Cost Profiler	application-cost-profiler
Amazon CloudWatch Application Insights	applicationinsights
AWS App Mesh	appmesh
Amazon WorkSpaces Applications	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	ajuste de escala automático
AWS Marketplace	aws-marketplace
AWS Backup	backup
AWS Batch	lote
Amazon Braket	braket

Serviço	Prefixo do serviço
AWS Budgets	orçamentos
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
Amazon CodeGuru Reviewer	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Notificações do	codestar-notifications
Identidade do Amazon Cognito	cognito-identity
Grupos de usuários do Amazon Cognito	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical

Serviço	Prefixo do serviço
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect Customer	conectar
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Amazon DocumentDB Elastic Clusters	docdb-elastic
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs

Serviço	Prefixo do serviço
Amazon Elastic Kubernetes Service	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon EMR em EKS (EMR Containers)	emr-containers
Amazon EMR Sem Servidor	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	eventos
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift Servers	gamelift
Amazon Location Service	geo
Amazon Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass

Serviço	Prefixo do serviço
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS Armazenamento de identidades do	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming for Apache Kafka	kafka

Serviço	Prefixo do serviço
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Linux Subscriptions Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	logs
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb

Serviço	Prefixo do serviço
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Migration Hub Strategy Recommendations	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organizações
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Perfis de clientes do Amazon Connect Customer	perfil
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Lixeira da	rbin
Amazon Relational Database Service	rds

Serviço	Prefixo do serviço
Amazon Redshift	redshift
API de dados do Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorador de recursos da AWS	resource-explorer-2
Grupos de recursos da AWS	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Roles Anywhere	rolesanywhere
Amazon Route 53	route53
Controles de recuperação do Amazon Route	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3
Amazon S3 on Outposts	s3-outposts
Recursos geoespaciais do Amazon SageMaker AI	sagemaker-geospatial
Savings Plans	savingsplans

Serviço	Prefixo do serviço
Esquemas do Amazon EventBridge	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub CSPM	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
SMS e serviço de voz do Amazon Pinpoint	sms-voice
AWS Snowball Edge	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
AWS Systems Manager para SAP	ssm-sap

Serviço	Prefixo do serviço
AWS Step Functions	estados
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	tag
Amazon Textract	textract
Amazon Timestream	timestream
AWS Telco Network Builder	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transferência
Amazon Translate	translate
ID de voz do Connect Customer	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Customer Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaços de trabalho
AWS X-Ray	xray

Ações para as informações acessadas pela última vez pela ação

A tabela a seguir lista as ações para as quais informações acessadas pela última vez pela ação estão disponíveis.

> **Important**
>
> A ação `iam:UpdateAccountName` será descontinuada em 22 de abril de 2026. Depois de 22 de abril de 2026, somente a permissão `account:PutAccountName` controlará o acesso à atualização do nome da conta. É altamente recomendável que você atualize todas as políticas de controle de serviços (SCPs) que controlam as atualizações do nome da conta para usar a permissão `account:PutAccountName`.

Prefixo do serviço	Ações do
access-analyzer	access-analyzer:ApplyArchiveRule
	access-analyzer:CancelPolicyGeneration
	access-analyzer:CheckAccessNotGranted
	access-analyzer:CheckNoNewAccess
	access-analyzer:CheckNoPublicAccess
	access-analyzer:CreateAccessPreview
	access-analyzer:CreateAnalyzer
	access-analyzer:CreateArchiveRule
	access-analyzer:DeleteAnalyzer
	access-analyzer:DeleteArchiveRule
	access-analyzer:GenerateFindingRecommendation
	access-analyzer:GetAccessPreview
	access-analyzer:GetAnalyzedResource

Prefixo do serviço	Ações do
	access-analyzer:GetAnalyzer
	access-analyzer:GetArchiveRule
	access-analyzer:GetFinding
	access-analyzer:GetFindingRecommendation
	access-analyzer:GetFindingsStatistics
	access-analyzer:GetGeneratedPolicy
	access-analyzer:ListAccessPreviewFindings
	access-analyzer:ListAccessPreviews
	access-analyzer:ListAnalyzedResources
	access-analyzer:ListAnalyzers
	access-analyzer:ListArchiveRules
	access-analyzer:ListFindings
	access-analyzer:ListPolicyGenerations
	access-analyzer:StartPolicyGeneration
	access-analyzer:StartResourceScan
	access-analyzer:UpdateAnalyzer
	access-analyzer:UpdateArchiveRule
	access-analyzer:UpdateFindings
	access-analyzer:ValidatePolicy

Prefixo do serviço	Ações do
conta	account:AcceptPrimaryEmailUpdate account:DeleteAlternateContact account:DisableRegion account:EnableRegion account:GetAccountInformation account:GetAlternateContact account:GetContactInformation account:GetGovCloudAccountInformation account:GetPrimaryEmail account:GetRegionOptStatus account:ListRegions account:PutAccountName account:PutAlternateContact account:PutContactInformation account:StartPrimaryEmailUpdate

Prefixo do serviço	Ações do
acm	acm:CreateAcmeEndpoint
	acm:DeleteCertificate
	acm:DescribeCertificate
	acm:ExportCertificate
	acm:GetAccountConfiguration
	acm:GetCertificate
	acm:ImportCertificate
	acm:ListAcmeEndpoints
	acm:ListCertificates
	acm:PutAccountConfiguration
	acm:RenewCertificate
	acm:RequestCertificate
	acm:ResendValidationEmail
	acm:SearchCertificates
	acm:UpdateCertificateOptions

Prefixo do serviço	Ações do
airflow	airflow:CreateCliToken airflow:CreateEnvironment airflow:CreateWebLoginToken airflow:DeleteEnvironment airflow:GetEnvironment airflow:ListEnvironments airflow:PublishMetrics airflow:UpdateEnvironment

Prefixo do serviço	Ações do
amplify	amplify:CreateApp
	amplify:CreateBackendEnvironment
	amplify:CreateBranch
	amplify:CreateDeployment
	amplify:CreateDomainAssociation
	amplify:CreateWebHook
	amplify:DeleteApp
	amplify:DeleteBackendEnvironment
	amplify:DeleteBranch
	amplify:DeleteDomainAssociation
	amplify:DeleteJob
	amplify:DeleteWebHook
	amplify:GenerateAccessLogs
	amplify:GetApp
	amplify:GetArtifactUrl
	amplify:GetBackendEnvironment
	amplify:GetBranch
	amplify:GetDomainAssociation
	amplify:GetJob
	amplify:GetWebHook
	amplify:ListApps

Prefixo do serviço	Ações do
	amplify:ListArtifacts
	amplify:ListBackendEnvironments
	amplify:ListBranches
	amplify:ListDomainAssociations
	amplify:ListJobs
	amplify:ListWebHooks
	amplify:StartDeployment
	amplify:StartJob
	amplify:StopJob
	amplify:UpdateApp
	amplify:UpdateBranch
	amplify:UpdateDomainAssociation
	amplify:UpdateWebHook

Prefixo do serviço	Ações do
amplifyuibuilder	amplifyuibuilder:CreateComponent amplifyuibuilder:CreateForm amplifyuibuilder:CreateTheme amplifyuibuilder:DeleteComponent amplifyuibuilder:DeleteForm amplifyuibuilder:DeleteTheme amplifyuibuilder:ExportComponents amplifyuibuilder:ExportThemes amplifyuibuilder:GetCodegenJob amplifyuibuilder:ListCodegenJobs amplifyuibuilder:ListComponents amplifyuibuilder:ListForms amplifyuibuilder:ListThemes amplifyuibuilder:ResetMetadataFlag amplifyuibuilder:StartCodegenJob amplifyuibuilder:UpdateComponent amplifyuibuilder:UpdateForm amplifyuibuilder:UpdateTheme

Prefixo do serviço	Ações do
app-integrations	app-integrations:CreateApplication
	app-integrations:CreateDataIntegration
	app-integrations:CreateDataIntegrationAssociation
	app-integrations:CreateEventIntegration
	app-integrations:DeleteApplication
	app-integrations:DeleteDataIntegration
	app-integrations:DeleteEventIntegration
	app-integrations:GetApplication
	app-integrations:GetDataIntegration
	app-integrations:GetEventIntegration
	app-integrations:ListApplicationAssociations
	app-integrations:ListApplications
	app-integrations:ListDataIntegrationAssociations
	app-integrations:ListDataIntegrations
	app-integrations:ListEventIntegrationAssociations
	app-integrations:ListEventIntegrations
	app-integrations:UpdateApplication
	app-integrations:UpdateDataIntegration
	app-integrations:UpdateDataIntegrationAssociation
	app-integrations:UpdateEventIntegration

Prefixo do serviço	Ações do
appconfig	appconfig:CreateApplication appconfig:CreateConfigurationProfile appconfig:CreateDeploymentStrategy appconfig:CreateEnvironment appconfig:CreateExtension appconfig:CreateExtensionAssociation appconfig:CreateHostedConfigurationVersion appconfig:DeleteApplication appconfig:DeleteConfigurationProfile appconfig:DeleteDeploymentStrategy appconfig:DeleteEnvironment appconfig:DeleteExtension appconfig:DeleteExtensionAssociation appconfig:DeleteHostedConfigurationVersion appconfig:GetAccountSettings appconfig:GetApplication appconfig:GetConfiguration appconfig:GetConfigurationProfile appconfig:GetDeployment appconfig:GetDeploymentStrategy appconfig:GetEnvironment

Prefixo do serviço	Ações do
	appconfig:GetExtension appconfig:GetExtensionAssociation appconfig:GetHostedConfigurationVersion appconfig:ListApplications appconfig:ListConfigurationProfiles appconfig:ListDeploymentStrategies appconfig:ListDeployments appconfig:ListEnvironments appconfig:ListExtensionAssociations appconfig:ListExtensions appconfig:ListHostedConfigurationVersions appconfig:StartDeployment appconfig:StopDeployment appconfig:UpdateAccountSettings appconfig:UpdateApplication appconfig:UpdateConfigurationProfile appconfig:UpdateDeploymentStrategy appconfig:UpdateEnvironment appconfig:UpdateExtension appconfig:UpdateExtensionAssociation appconfig:ValidateConfiguration

Prefixo do serviço	Ações do
appflow	appflow:CancelFlowExecutions
	appflow:CreateConnectorProfile
	appflow:CreateFlow
	appflow:DeleteConnectorProfile
	appflow:DeleteFlow
	appflow:DescribeConnector
	appflow:DescribeConnectorEntity
	appflow:DescribeConnectorProfiles
	appflow:DescribeConnectors
	appflow:DescribeFlow
	appflow:DescribeFlowExecutionRecords
	appflow:ListConnectorEntities
	appflow:ListConnectors
	appflow:ListFlows
	appflow:RegisterConnector
	appflow:ResetConnectorMetadataCache
	appflow:StartFlow
	appflow:StopFlow
	appflow:UnRegisterConnector
	appflow:UpdateConnectorProfile
	appflow:UpdateConnectorRegistration

Prefixo do serviço	Ações do
	appflow:UpdateFlow

Prefixo do serviço	Ações do
applicationinsights	applicationinsights:AddWorkload applicationinsights:CreateApplication applicationinsights:CreateComponent applicationinsights:CreateLogPattern applicationinsights:DeleteApplication applicationinsights:DeleteComponent applicationinsights:DeleteLogPattern applicationinsights:DescribeApplication applicationinsights:DescribeComponent applicationinsights:DescribeComponentConfiguration applicationinsights:DescribeComponentConfigurationRecommendation applicationinsights:DescribeLogPattern applicationinsights:DescribeObservation applicationinsights:DescribeProblem applicationinsights:DescribeProblemObservations applicationinsights:DescribeWorkload applicationinsights:ListApplications applicationinsights:ListComponents applicationinsights:ListConfigurationHistory applicationinsights:ListLogPatternSets

Prefixo do serviço	Ações do
	applicationinsights:ListLogPatterns applicationinsights:ListProblems applicationinsights:ListWorkloads applicationinsights:RemoveWorkload applicationinsights:UpdateApplication applicationinsights:UpdateComponent applicationinsights:UpdateComponentConfiguration applicationinsights:UpdateLogPattern applicationinsights:UpdateWorkload

Prefixo do serviço	Ações do
appmesh	appmesh:CreateGatewayRoute
	appmesh:CreateMesh
	appmesh:CreateRoute
	appmesh:CreateVirtualGateway
	appmesh:CreateVirtualNode
	appmesh:CreateVirtualRouter
	appmesh:CreateVirtualService
	appmesh:DeleteGatewayRoute
	appmesh:DeleteMesh
	appmesh:DeleteRoute
	appmesh:DeleteVirtualGateway
	appmesh:DeleteVirtualNode
	appmesh:DeleteVirtualRouter
	appmesh:DeleteVirtualService
	appmesh:DescribeGatewayRoute
	appmesh:DescribeMesh
	appmesh:DescribeRoute
	appmesh:DescribeVirtualGateway
	appmesh:DescribeVirtualNode
	appmesh:DescribeVirtualRouter
	appmesh:DescribeVirtualService

Prefixo do serviço	Ações do
	appmesh:ListGatewayRoutes
	appmesh:ListMeshes
	appmesh:ListRoutes
	appmesh:ListVirtualGateways
	appmesh:ListVirtualNodes
	appmesh:ListVirtualRouters
	appmesh:ListVirtualServices
	appmesh:StreamAggregatedResources
	appmesh:UpdateGatewayRoute
	appmesh:UpdateMesh
	appmesh:UpdateRoute
	appmesh:UpdateVirtualGateway
	appmesh:UpdateVirtualNode
	appmesh:UpdateVirtualRouter
	appmesh:UpdateVirtualService

Prefixo do serviço	Ações do
appstream	appstream:AssociateAppBlockBuilderAppBlock appstream:AssociateApplicationFleet appstream:AssociateApplicationToEntitlement appstream:AssociateFleet appstream:AssociateSoftwareToImageBuilder appstream:BatchAssociateUserStack appstream:BatchDisassociateUserStack appstream:CopyImage appstream:CreateAppBlock appstream:CreateAppBlockBuilder appstream:CreateAppBlockBuilderStreamingURL appstream:CreateApplication appstream:CreateDirectoryConfig appstream:CreateEntitlement appstream:CreateFleet appstream:CreateImageBuilder appstream:CreateImageBuilderStreamingURL appstream:CreateStack appstream:CreateStreamingURL appstream:CreateThemeForStack appstream:CreateUpdatedImage

Prefixo do serviço	Ações do
	appstream:CreateUsageReportSubscription appstream:CreateUser appstream:DeleteAppBlock appstream:DeleteAppBlockBuilder appstream:DeleteApplication appstream:DeleteDirectoryConfig appstream:DeleteEntitlement appstream:DeleteFleet appstream:DeleteImage appstream:DeleteImageBuilder appstream:DeleteImagePermissions appstream:DeleteStack appstream:DeleteThemeForStack appstream:DeleteUsageReportSubscription appstream:DeleteUser appstream:DescribeAppBlockBuilderAppBlockAssociations appstream:DescribeAppBlockBuilders appstream:DescribeAppBlocks appstream:DescribeAppLicenseUsage appstream:DescribeApplicationFleetAssociations appstream:DescribeApplications

Prefixo do serviço	Ações do
	appstream:DescribeDirectoryConfigs appstream:DescribeEntitlements appstream:DescribeFleets appstream:DescribeImageBuilders appstream:DescribeImagePermissions appstream:DescribeImages appstream:DescribeSessions appstream:DescribeStacks appstream:DescribeThemeForStack appstream:DescribeUsageReportSubscriptions appstream:DescribeUserStackAssociations appstream:DescribeUsers appstream:DisableUser appstream:DisassociateAppBlockBuilderAppBlock appstream:DisassociateApplicationFleet appstream:DisassociateApplicationFromEntitlement appstream:DisassociateFleet appstream:DisassociateSoftwareFromImageBuilder appstream:EnableUser appstream:ExpireSession appstream:GetExportImageTask

Prefixo do serviço	Ações do
	appstream:ListAssociatedFleets appstream:ListAssociatedStacks appstream:ListEntitledApplications appstream:ListExportImageTasks appstream:StartAppBlockBuilder appstream:StartFleet appstream:StartImageBuilder appstream:StartSoftwareDeploymentToImageBuilder appstream:StopAppBlockBuilder appstream:StopFleet appstream:StopImageBuilder appstream:UpdateAppBlockBuilder appstream:UpdateApplication appstream:UpdateDirectoryConfig appstream:UpdateEntitlement appstream:UpdateFleet appstream:UpdateImagePermissions appstream:UpdateStack appstream:UpdateThemeForStack

Prefixo do serviço	Ações do
appsync	appsync:AssociateApi
	appsync:AssociateMergedGraphqlApi
	appsync:AssociateSourceGraphqlApi
	appsync:CreateApi
	appsync:CreateApiCache
	appsync:CreateApiKey
	appsync:CreateChannelNamespace
	appsync:CreateDataSource
	appsync:CreateDomainName
	appsync:CreateFunction
	appsync:CreateGraphqlApi
	appsync:CreateResolver
	appsync:CreateType
	appsync:DeleteApi
	appsync:DeleteApiCache
	appsync:DeleteApiKey
	appsync:DeleteChannelNamespace
	appsync:DeleteDataSource
	appsync:DeleteDomainName
	appsync:DeleteFunction
	appsync:DeleteGraphqlApi

Prefixo do serviço	Ações do
	appsync:DeleteResolver
	appsync:DeleteType
	appsync:DisassociateApi
	appsync:DisassociateMergedGraphqlApi
	appsync:DisassociateSourceGraphqlApi
	appsync:EvaluateCode
	appsync:EvaluateMappingTemplate
	appsync:FlushApiCache
	appsync:GetApi
	appsync:GetApiAssociation
	appsync:GetApiCache
	appsync:GetChannelNamespace
	appsync:GetDataSource
	appsync:GetDataSourceIntrospection
	appsync:GetDomainName
	appsync:GetFunction
	appsync:GetGraphqlApi
	appsync:GetGraphqlApiEnvironmentVariables
	appsync:GetIntrospectionSchema
	appsync:GetResolver
	appsync:GetSchemaCreationStatus

Prefixo do serviço	Ações do
	appsync:GetSourceApiAssociation
	appsync:GetType
	appsync:ListApiKeys
	appsync:ListApis
	appsync:ListChannelNamespaces
	appsync:ListDataSources
	appsync:ListDomainNames
	appsync:ListFunctions
	appsync:ListGraphqlApis
	appsync:ListResolvers
	appsync:ListResolversByFunction
	appsync:ListSourceApiAssociations
	appsync:ListTypes
	appsync:ListTypesByAssociation
	appsync:PutGraphqlApiEnvironmentVariables
	appsync:StartDataSourceIntrospection
	appsync:StartSchemaCreation
	appsync:StartSchemaMerge
	appsync:UpdateApi
	appsync:UpdateApiCache
	appsync:UpdateApiKey

Prefixo do serviço	Ações do
	appsync:UpdateChannelNamespace
	appsync:UpdateDataSource
	appsync:UpdateDomainName
	appsync:UpdateFunction
	appsync:UpdateGraphqlApi
	appsync:UpdateResolver
	appsync:UpdateSourceApiAssociation
	appsync:UpdateType

Prefixo do serviço	Ações do
aps	aps:CreateAlertManagerDefinition aps:CreateAnomalyDetector aps:CreateLoggingConfiguration aps:CreateQueryLoggingConfiguration aps:CreateRuleGroupsNamespace aps:CreateWorkspace aps:DeleteAlertManagerDefinition aps:DeleteAnomalyDetector aps:DeleteLoggingConfiguration aps:DeleteQueryLoggingConfiguration aps:DeleteResourcePolicy aps:DeleteRuleGroupsNamespace aps:DeleteScraper aps:DeleteScraperLoggingConfiguration aps:DeleteWorkspace aps:DescribeAlertManagerDefinition aps:DescribeAnomalyDetector aps:DescribeLoggingConfiguration aps:DescribeQueryLoggingConfiguration aps:DescribeResourcePolicy aps:DescribeRuleGroupsNamespace

Prefixo do serviço	Ações do
	aps:DescribeScraper aps:DescribeScraperLoggingConfiguration aps:DescribeWorkspace aps:DescribeWorkspaceConfiguration aps:GetDefaultScraperConfiguration aps:ListAnomalyDetectors aps:ListRuleGroupsNamespaces aps:ListScrapers aps:ListWorkspaces aps:PutAlertManagerDefinition aps:PutAnomalyDetector aps:PutResourcePolicy aps:PutRuleGroupsNamespace aps:UpdateLoggingConfiguration aps:UpdateQueryLoggingConfiguration aps:UpdateScraper aps:UpdateScraperLoggingConfiguration aps:UpdateWorkspaceAlias aps:UpdateWorkspaceConfiguration

Prefixo do serviço	Ações do
athena	athena:BatchGetNamedQuery
	athena:BatchGetPreparedStatement
	athena:BatchGetQueryExecution
	athena:CancelCapacityReservation
	athena:CreateCapacityReservation
	athena:CreateDataCatalog
	athena:CreateNamedQuery
	athena:CreateNotebook
	athena:CreatePreparedStatement
	athena:CreatePresignedNotebookUrl
	athena:CreateWorkGroup
	athena:DeleteCapacityReservation
	athena:DeleteDataCatalog
	athena:DeleteNamedQuery
	athena:DeleteNotebook
	athena:DeletePreparedStatement
	athena:DeleteWorkGroup
	athena:ExportNotebook
	athena:GetCalculationExecution
	athena:GetCalculationExecutionCode
	athena:GetCalculationExecutionStatus

Prefixo do serviço	Ações do
	athena:GetCapacityAssignmentConfiguration
	athena:GetCapacityReservation
	athena:GetDataCatalog
	athena:GetDatabase
	athena:GetNamedQuery
	athena:GetNotebookMetadata
	athena:GetPreparedStatement
	athena:GetQueryExecution
	athena:GetQueryResults
	athena:GetQueryResultsStream
	athena:GetQueryRuntimeStatistics
	athena:GetResourceDashboard
	athena:GetSession
	athena:GetSessionEndpoint
	athena:GetSessionStatus
	athena:GetTableMetadata
	athena:GetWorkGroup
	athena:ImportNotebook
	athena:ListApplicationDPUSizes
	athena:ListCalculationExecutions
	athena:ListCapacityReservations

Prefixo do serviço	Ações do
	athena:ListDataCatalogs
	athena:ListDatabases
	athena:ListEngineVersions
	athena:ListExecutors
	athena:ListNamedQueries
	athena:ListNotebookMetadata
	athena:ListNotebookSessions
	athena:ListPreparedStatements
	athena:ListQueryExecutions
	athena:ListSessions
	athena:ListTableMetadata
	athena:ListWorkGroups
	athena:PutCapacityAssignmentConfiguration
	athena:StartCalculationExecution
	athena:StartQueryExecution
	athena:StartSession
	athena:StopCalculationExecution
	athena:StopQueryExecution
	athena:TerminateSession
	athena:UpdateCapacityReservation
	athena:UpdateDataCatalog

Prefixo do serviço	Ações do
	athena:UpdateNamedQuery
	athena:UpdateNotebook
	athena:UpdateNotebookMetadata
	athena:UpdatePreparedStatement
	athena:UpdateWorkGroup

Prefixo do serviço	Ações do
auditmanager	auditmanager:AssociateAssessmentReportEvidenceFolder auditmanager:BatchAssociateAssessmentReportEvidence auditmanager:BatchCreateDelegationByAssessment auditmanager:BatchDeleteDelegationByAssessment auditmanager:BatchDisassociateAssessmentReportEvidence auditmanager:BatchImportEvidenceToAssessmentControl auditmanager:CreateAssessment auditmanager:CreateAssessmentFramework auditmanager:CreateAssessmentReport auditmanager:CreateControl auditmanager:DeleteAssessment auditmanager:DeleteAssessmentFramework auditmanager:DeleteAssessmentFrameworkShare auditmanager:DeleteAssessmentReport auditmanager:DeleteControl auditmanager:DeregisterAccount auditmanager:DeregisterOrganizationAdminAccount auditmanager:DisassociateAssessmentReportEvidenceFolder auditmanager:GetAccountStatus auditmanager:GetAssessment auditmanager:GetAssessmentFramework

Prefixo do serviço	Ações do
	auditmanager:GetAssessmentReportUrl auditmanager:GetChangeLogs auditmanager:GetControl auditmanager:GetDelegations auditmanager:GetEvidence auditmanager:GetEvidenceByEvidenceFolder auditmanager:GetEvidenceFileUploadUrl auditmanager:GetEvidenceFolder auditmanager:GetEvidenceFoldersByAssessment auditmanager:GetEvidenceFoldersByAssessmentControl auditmanager:GetInsights auditmanager:GetInsightsByAssessment auditmanager:GetOrganizationAdminAccount auditmanager:GetServicesInScope auditmanager:GetSettings auditmanager:ListAssessmentControlInsightsByControlDomain auditmanager:ListAssessmentFrameworkShareRequests auditmanager:ListAssessmentFrameworks auditmanager:ListAssessmentReports auditmanager:ListAssessments auditmanager:ListControlDomainInsights

Prefixo do serviço	Ações do
	auditmanager:ListControlDomainInsightsByAssessment auditmanager:ListControlInsightsByControlDomain auditmanager:ListControls auditmanager:ListKeywordsForDataSource auditmanager:ListNotifications auditmanager:RegisterAccount auditmanager:RegisterOrganizationAdminAccount auditmanager:StartAssessmentFrameworkShare auditmanager:UpdateAssessment auditmanager:UpdateAssessmentControl auditmanager:UpdateAssessmentControlSetStatus auditmanager:UpdateAssessmentFramework auditmanager:UpdateAssessmentFrameworkShare auditmanager:UpdateAssessmentStatus auditmanager:UpdateControl auditmanager:UpdateSettings auditmanager:ValidateAssessmentReportIntegrity

Prefixo do serviço	Ações do
ajuste de escala automático	autoscaling:AttachInstances
	autoscaling:AttachLoadBalancerTargetGroups
	autoscaling:AttachLoadBalancers
	autoscaling:AttachTrafficSources
	autoscaling:BatchDeleteScheduledAction
	autoscaling:BatchPutScheduledUpdateGroupAction
	autoscaling:CancelInstanceRefresh
	autoscaling:CompleteLifecycleAction
	autoscaling:CreateAutoScalingGroup
	autoscaling:CreateLaunchConfiguration
	autoscaling:DeleteAutoScalingGroup
	autoscaling:DeleteLaunchConfiguration
	autoscaling:DeleteLifecycleHook
	autoscaling:DeleteNotificationConfiguration
	autoscaling:DeletePolicy
	autoscaling:DeleteScheduledAction
	autoscaling:DeleteWarmPool
	autoscaling:DescribeAccountLimits
	autoscaling:DescribeAdjustmentTypes
	autoscaling:DescribeAutoScalingGroups
	autoscaling:DescribeAutoScalingInstances

Prefixo do serviço	Ações do
	autoscaling:DescribeAutoScalingNotificationTypes
	autoscaling:DescribeInstanceRefreshes
	autoscaling:DescribeLaunchConfigurations
	autoscaling:DescribeLifecycleHookTypes
	autoscaling:DescribeLifecycleHooks
	autoscaling:DescribeLoadBalancerTargetGroups
	autoscaling:DescribeLoadBalancers
	autoscaling:DescribeMetricCollectionTypes
	autoscaling:DescribeNotificationConfigurations
	autoscaling:DescribePolicies
	autoscaling:DescribeScalingActivities
	autoscaling:DescribeScalingProcessTypes
	autoscaling:DescribeScheduledActions
	autoscaling:DescribeTerminationPolicyTypes
	autoscaling:DescribeTrafficSources
	autoscaling:DescribeWarmPool
	autoscaling:DetachInstances
	autoscaling:DetachLoadBalancerTargetGroups
	autoscaling:DetachLoadBalancers
	autoscaling:DetachTrafficSources
	autoscaling:DisableMetricsCollection

Prefixo do serviço	Ações do
	autoscaling:EnableMetricsCollection autoscaling:EnterStandby autoscaling:ExecutePolicy autoscaling:ExitStandby autoscaling:GetPredictiveScalingForecast autoscaling:PutLifecycleHook autoscaling:PutNotificationConfiguration autoscaling:PutScalingPolicy autoscaling:PutScheduledUpdateGroupAction autoscaling:PutWarmPool autoscaling:RecordLifecycleActionHeartbeat autoscaling:ResumeProcesses autoscaling:RollbackInstanceRefresh autoscaling:SetDesiredCapacity autoscaling:SetInstanceHealth autoscaling:SetInstanceProtection autoscaling:StartInstanceRefresh autoscaling:SuspendProcesses autoscaling:TerminateInstanceInAutoScalingGroup autoscaling:UpdateAutoScalingGroup
aws-marketplace	aws-marketplace:GetEntitlements

Prefixo do serviço	Ações do
backup	backup:CancelLegalHold
	backup:CreateBackupPlan
	backup:CreateBackupSelection
	backup:CreateBackupVault
	backup:CreateFramework
	backup:CreateLegalHold
	backup:CreateLogicallyAirGappedBackupVault
	backup:CreateReportPlan
	backup:CreateRestoreAccessBackupVault
	backup:CreateRestoreTestingPlan
	backup:CreateRestoreTestingSelection
	backup:CreateTieringConfiguration
	backup:DeleteBackupPlan
	backup:DeleteBackupSelection
	backup:DeleteBackupVault
	backup:DeleteBackupVaultAccessPolicy
	backup:DeleteBackupVaultLockConfiguration
	backup:DeleteBackupVaultNotifications
	backup:DeleteFramework
	backup:DeleteRecoveryPoint
	backup:DeleteReportPlan

Prefixo do serviço	Ações do
	backup:DeleteRestoreTestingPlan
	backup:DeleteRestoreTestingSelection
	backup:DeleteTieringConfiguration
	backup:DescribeBackupJob
	backup:DescribeBackupVault
	backup:DescribeCopyJob
	backup:DescribeFramework
	backup:DescribeGlobalSettings
	backup:DescribeProtectedResource
	backup:DescribeRecoveryPoint
	backup:DescribeRegionSettings
	backup:DescribeReportJob
	backup:DescribeReportPlan
	backup:DescribeRestoreJob
	backup:DescribeScanJob
	backup:DisassociateRecoveryPoint
	backup:DisassociateRecoveryPointFromParent
	backup:ExportBackupPlanTemplate
	backup:GetBackupPlan
	backup:GetBackupPlanFromJSON
	backup:GetBackupPlanFromTemplate

Prefixo do serviço	Ações do
	backup:GetBackupSelection
	backup:GetBackupVaultAccessPolicy
	backup:GetBackupVaultNotifications
	backup:GetLegalHold
	backup:GetRecoveryPointRestoreMetadata
	backup:GetRestoreJobMetadata
	backup:GetRestoreTestingInferredMetadata
	backup:GetRestoreTestingPlan
	backup:GetRestoreTestingSelection
	backup:GetSupportedResourceTypes
	backup:GetTieringConfiguration
	backup:ListBackupJobSummaries
	backup:ListBackupJobs
	backup:ListBackupPlanTemplates
	backup:ListBackupPlanVersions
	backup:ListBackupPlans
	backup:ListBackupSelections
	backup:ListBackupVaults
	backup:ListCopyJobSummaries
	backup:ListCopyJobs
	backup:ListFrameworks

Prefixo do serviço	Ações do
	backup:ListIndexedRecoveryPoints
	backup:ListLegalHolds
	backup:ListProtectedResources
	backup:ListRecoveryPointsByBackupVault
	backup:ListRecoveryPointsByLegalHold
	backup:ListRecoveryPointsByResource
	backup:ListReportJobs
	backup:ListReportPlans
	backup:ListRestoreAccessBackupVaults
	backup:ListRestoreJobSummaries
	backup:ListRestoreJobs
	backup:ListRestoreJobsByProtectedResource
	backup:ListRestoreTestingPlans
	backup:ListRestoreTestingSelections
	backup:ListScanJobSummaries
	backup:ListScanJobs
	backup:ListTieringConfigurations
	backup:PutBackupVaultAccessPolicy
	backup:PutBackupVaultLockConfiguration
	backup:PutBackupVaultNotifications
	backup:PutRestoreValidationResult

Prefixo do serviço	Ações do
	backup:StartBackupJob
	backup:StartCopyJob
	backup:StartReportJob
	backup:StartRestoreJob
	backup:StopBackupJob
	backup:UpdateBackupPlan
	backup:UpdateFramework
	backup:UpdateGlobalSettings
	backup:UpdateRecoveryPointLifecycle
	backup:UpdateRegionSettings
	backup:UpdateReportPlan
	backup:UpdateRestoreTestingPlan
	backup:UpdateRestoreTestingSelection
	backup:UpdateTieringConfiguration

Prefixo do serviço	Ações do
lote	batch:CancelJob batch:CreateComputeEnvironment batch:CreateConsumableResource batch:CreateJobQueue batch:CreateQuotaShare batch:CreateSchedulingPolicy batch:CreateServiceEnvironment batch:DeleteComputeEnvironment batch:DeleteConsumableResource batch:DeleteJobQueue batch:DeleteQuotaShare batch:DeleteSchedulingPolicy batch:DeleteServiceEnvironment batch:DeregisterJobDefinition batch:DescribeComputeEnvironments batch:DescribeConsumableResource batch:DescribeJobDefinitions batch:DescribeJobQueues batch:DescribeJobs batch:DescribeQuotaShare batch:DescribeSchedulingPolicies

Prefixo do serviço	Ações do
	batch:DescribeServiceEnvironments batch:DescribeServiceJob batch:GetJobQueueSnapshot batch:ListConsumableResources batch:ListJobs batch:ListJobsByConsumableResource batch:ListQuotaShares batch:ListSchedulingPolicies batch:ListServiceJobs batch:RegisterJobDefinition batch:SubmitJob batch:SubmitServiceJob batch:TerminateJob batch:TerminateServiceJob batch:UpdateComputeEnvironment batch:UpdateConsumableResource batch:UpdateJobQueue batch:UpdateQuotaShare batch:UpdateSchedulingPolicy batch:UpdateServiceEnvironment batch:UpdateServiceJob

Prefixo do serviço	Ações do
braket	braket:CancelJob braket:CancelQuantumTask braket:CreateJob braket:CreateQuantumTask braket:CreateSpendingLimit braket:GetDevice braket:GetJob braket:GetQuantumTask braket:SearchDevices braket:SearchJobs braket:SearchQuantumTasks braket:SearchSpendingLimits

Prefixo do serviço	Ações do
orçamentos	budgets:CreateBudgetAction budgets:DeleteBudgetAction budgets:DescribeBudgetAction budgets:DescribeBudgetActionHistories budgets:DescribeBudgetActionsForAccount budgets:DescribeBudgetActionsForBudget budgets:ExecuteBudgetAction budgets:ModifyBudget budgets:UpdateBudgetAction budgets:ViewBudget
cloud9	cloud9:CreateEnvironmentEC2 cloud9:CreateEnvironmentMembership cloud9:DeleteEnvironment cloud9:DeleteEnvironmentMembership cloud9:DescribeEnvironmentMemberships cloud9:DescribeEnvironmentStatus cloud9:DescribeEnvironments cloud9:ListEnvironments cloud9:UpdateEnvironment cloud9:UpdateEnvironmentMembership

Prefixo do serviço	Ações do
cloudformation	cloudformation:BatchDescribeTypeConfigurations cloudformation:CancelUpdateStack cloudformation:ContinueUpdateRollback cloudformation:CreateChangeSet cloudformation:CreateGeneratedTemplate cloudformation:CreateStack cloudformation:CreateStackInstances cloudformation:CreateStackSet cloudformation:DeactivateType cloudformation:DeleteChangeSet cloudformation:DeleteGeneratedTemplate cloudformation:DeleteStack cloudformation:DeleteStackInstances cloudformation:DeleteStackSet cloudformation:DeregisterType cloudformation:DescribeAccountLimits cloudformation:DescribeChangeSet cloudformation:DescribeChangeSetHooks cloudformation:DescribeEvents cloudformation:DescribeGeneratedTemplate cloudformation:DescribeOrganizationsAccess

Prefixo do serviço	Ações do
	cloudformation:DescribePublisher cloudformation:DescribeResourceScan cloudformation:DescribeStackDriftDetectionStatus cloudformation:DescribeStackEvents cloudformation:DescribeStackInstance cloudformation:DescribeStackResource cloudformation:DescribeStackResourceDrifts cloudformation:DescribeStackResources cloudformation:DescribeStackSet cloudformation:DescribeStackSetOperation cloudformation:DescribeStacks cloudformation:DescribeType cloudformation:DescribeTypeRegistration cloudformation:DetectStackDrift cloudformation:DetectStackResourceDrift cloudformation:DetectStackSetDrift cloudformation:EstimateTemplateCost cloudformation:ExecuteChangeSet cloudformation:GetGeneratedTemplate cloudformation:GetHookResult cloudformation:GetStackPolicy

Prefixo do serviço	Ações do
	cloudformation:GetTemplate cloudformation:GetTemplateSummary cloudformation:ImportStacksToStackSet cloudformation:ListChangeSets cloudformation:ListExports cloudformation:ListGeneratedTemplates cloudformation:ListHookResults cloudformation:ListImports cloudformation:ListResourceScanRelatedResources cloudformation:ListResourceScanResources cloudformation:ListResourceScans cloudformation:ListStackInstanceResourceDrifts cloudformation:ListStackInstances cloudformation:ListStackRefactors cloudformation:ListStackResources cloudformation:ListStackSetAutoDeploymentTargets cloudformation:ListStackSetOperationResults cloudformation:ListStackSetOperations cloudformation:ListStackSets cloudformation:ListTypeRegistrations cloudformation:ListTypeVersions

Prefixo do serviço	Ações do
	cloudformation:ListTypes cloudformation:PublishType cloudformation:RecordHandlerProgress cloudformation:RegisterPublisher cloudformation:RegisterType cloudformation:RollbackStack cloudformation:SetStackPolicy cloudformation:SetTypeConfiguration cloudformation:SetTypeDefaultVersion cloudformation:SignalResource cloudformation:StartResourceScan cloudformation:StopStackSetOperation cloudformation:TestType cloudformation:UpdateGeneratedTemplate cloudformation:UpdateStack cloudformation:UpdateStackInstances cloudformation:UpdateStackSet cloudformation:UpdateTerminationProtection cloudformation:ValidateTemplate

Prefixo do serviço	Ações do
cloudfront	cloudfront:AssociateAlias
	cloudfront:AssociateDistributionTenantWebACL
	cloudfront:AssociateDistributionWebACL
	cloudfront:CreateCachePolicy
	cloudfront:CreateCloudFrontOriginAccessIdentity
	cloudfront:CreateConnectionFunction
	cloudfront:CreateContinuousDeploymentPolicy
	cloudfront:CreateDistributionTenant
	cloudfront:CreateFieldLevelEncryptionConfig
	cloudfront:CreateFieldLevelEncryptionProfile
	cloudfront:CreateFunction
	cloudfront:CreateInvalidation
	cloudfront:CreateKeyGroup
	cloudfront:CreateKeyValueStore
	cloudfront:CreateMonitoringSubscription
	cloudfront:CreateOriginAccessControl
	cloudfront:CreateOriginRequestPolicy
	cloudfront:CreatePublicKey
	cloudfront:CreateRealtimeLogConfig
	cloudfront:CreateResponseHeadersPolicy
	cloudfront:CreateTrustStore

Prefixo do serviço	Ações do
	cloudfront:DeleteAnycastIpList
	cloudfront:DeleteCachePolicy
	cloudfront:DeleteCloudFrontOriginAccessIdentity
	cloudfront:DeleteConnectionFunction
	cloudfront:DeleteConnectionGroup
	cloudfront:DeleteContinuousDeploymentPolicy
	cloudfront:DeleteDistribution
	cloudfront:DeleteDistributionTenant
	cloudfront:DeleteFieldLevelEncryptionConfig
	cloudfront:DeleteFieldLevelEncryptionProfile
	cloudfront:DeleteFunction
	cloudfront:DeleteKeyGroup
	cloudfront:DeleteKeyValueStore
	cloudfront:DeleteMonitoringSubscription
	cloudfront:DeleteOriginAccessControl
	cloudfront:DeleteOriginRequestPolicy
	cloudfront:DeletePublicKey
	cloudfront:DeleteRealtimeLogConfig
	cloudfront:DeleteResponseHeadersPolicy
	cloudfront:DeleteStreamingDistribution
	cloudfront:DeleteTrustStore

Prefixo do serviço	Ações do
	cloudfront:DeleteVpcOrigin
	cloudfront:DescribeFunction
	cloudfront:DescribeKeyValueStore
	cloudfront:DisassociateDistributionTenantWebACL
	cloudfront:DisassociateDistributionWebACL
	cloudfront:GetAnycastIpList
	cloudfront:GetCachePolicy
	cloudfront:GetCachePolicyConfig
	cloudfront:GetCloudFrontOriginAccessIdentity
	cloudfront:GetCloudFrontOriginAccessIdentityConfig
	cloudfront:GetContinuousDeploymentPolicy
	cloudfront:GetContinuousDeploymentPolicyConfig
	cloudfront:GetDistributionConfig
	cloudfront:GetFieldLevelEncryption
	cloudfront:GetFieldLevelEncryptionConfig
	cloudfront:GetFieldLevelEncryptionProfile
	cloudfront:GetFieldLevelEncryptionProfileConfig
	cloudfront:GetFunction
	cloudfront:GetInvalidation
	cloudfront:GetInvalidationForDistributionTenant
	cloudfront:GetKeyGroup

Prefixo do serviço	Ações do
	cloudfront:GetKeyGroupConfig
	cloudfront:GetMonitoringSubscription
	cloudfront:GetOriginAccessControl
	cloudfront:GetOriginAccessControlConfig
	cloudfront:GetOriginRequestPolicy
	cloudfront:GetOriginRequestPolicyConfig
	cloudfront:GetPublicKey
	cloudfront:GetPublicKeyConfig
	cloudfront:GetRealtimeLogConfig
	cloudfront:GetResponseHeadersPolicy
	cloudfront:GetResponseHeadersPolicyConfig
	cloudfront:GetStreamingDistribution
	cloudfront:GetStreamingDistributionConfig
	cloudfront:GetVpcOrigin
	cloudfront:ListAnycastIpLists
	cloudfront:ListCachePolicies
	cloudfront:ListCloudFrontOriginAccessIdentities
	cloudfront:ListConflictingAliases
	cloudfront:ListConnectionFunctions
	cloudfront:ListConnectionGroups
	cloudfront:ListContinuousDeploymentPolicies

Prefixo do serviço	Ações do
	cloudfront:ListDistributionTenants
	cloudfront:ListDistributionTenantsByCustomization
	cloudfront:ListDistributions
	cloudfront:ListDistributionsByAnycastIpListId
	cloudfront:ListDistributionsByCachePolicyId
	cloudfront:ListDistributionsByConnectionMode
	cloudfront:ListDistributionsByKeyGroup
	cloudfront:ListDistributionsByOriginRequestPolicyId
	cloudfront:ListDistributionsByRealtimeLogConfig
	cloudfront:ListDistributionsByResponseHeadersPolicyId
	cloudfront:ListDistributionsByVpcOriginId
	cloudfront:ListDistributionsByWebACLId
	cloudfront:ListFieldLevelEncryptionConfigs
	cloudfront:ListFieldLevelEncryptionProfiles
	cloudfront:ListFunctions
	cloudfront:ListInvalidations
	cloudfront:ListInvalidationsForDistributionTenant
	cloudfront:ListKeyGroups
	cloudfront:ListKeyValueStores
	cloudfront:ListOriginAccessControls
	cloudfront:ListOriginRequestPolicies

Prefixo do serviço	Ações do
	cloudfront:ListPublicKeys
	cloudfront:ListRealtimeLogConfigs
	cloudfront:ListResponseHeadersPolicies
	cloudfront:ListStreamingDistributions
	cloudfront:ListTrustStores
	cloudfront:PublishConnectionFunction
	cloudfront:PublishFunction
	cloudfront:TestConnectionFunction
	cloudfront:TestFunction
	cloudfront:UpdateAnycastIpList
	cloudfront:UpdateCachePolicy
	cloudfront:UpdateCloudFrontOriginAccessIdentity
	cloudfront:UpdateConnectionFunction
	cloudfront:UpdateConnectionGroup
	cloudfront:UpdateContinuousDeploymentPolicy
	cloudfront:UpdateDistribution
	cloudfront:UpdateDistributionTenant
	cloudfront:UpdateFieldLevelEncryptionConfig
	cloudfront:UpdateFieldLevelEncryptionProfile
	cloudfront:UpdateFunction
	cloudfront:UpdateKeyGroup

Prefixo do serviço	Ações do
	cloudfront:UpdateKeyValueStore cloudfront:UpdateOriginAccessControl cloudfront:UpdateOriginRequestPolicy cloudfront:UpdatePublicKey cloudfront:UpdateRealtimeLogConfig cloudfront:UpdateResponseHeadersPolicy cloudfront:UpdateTrustStore
cloudhsm	cloudhsm:CreateHsm cloudhsm:DeleteBackup cloudhsm:DeleteHsm cloudhsm:DeleteResourcePolicy cloudhsm:DescribeBackups cloudhsm:DescribeClusters cloudhsm:GetResourcePolicy cloudhsm:InitializeCluster cloudhsm:ModifyBackupAttributes cloudhsm:ModifyCluster cloudhsm:PutResourcePolicy cloudhsm:RestoreBackup

Prefixo do serviço	Ações do
cloudsearch	cloudsearch:BuildSuggesters
	cloudsearch:CreateDomain
	cloudsearch:DefineAnalysisScheme
	cloudsearch:DefineExpression
	cloudsearch:DefineIndexField
	cloudsearch:DefineSuggester
	cloudsearch:DeleteAnalysisScheme
	cloudsearch:DeleteDomain
	cloudsearch:DeleteExpression
	cloudsearch:DeleteIndexField
	cloudsearch:DeleteSuggester
	cloudsearch:DescribeAnalysisSchemes
	cloudsearch:DescribeAvailabilityOptions
	cloudsearch:DescribeDomainEndpointOptions
	cloudsearch:DescribeDomains
	cloudsearch:DescribeExpressions
	cloudsearch:DescribeIndexFields
	cloudsearch:DescribeScalingParameters
	cloudsearch:DescribeServiceAccessPolicies
	cloudsearch:DescribeSuggesters
	cloudsearch:IndexDocuments

Prefixo do serviço	Ações do
	cloudsearch:ListDomainNames
	cloudsearch:UpdateAvailabilityOptions
	cloudsearch:UpdateDomainEndpointOptions
	cloudsearch:UpdateScalingParameters
	cloudsearch:UpdateServiceAccessPolicies

Prefixo do serviço	Ações do
cloudtrail	cloudtrail:CancelQuery cloudtrail:CreateChannel cloudtrail:CreateDashboard cloudtrail:CreateEventDataStore cloudtrail:CreateTrail cloudtrail:DeleteChannel cloudtrail:DeleteDashboard cloudtrail:DeleteEventDataStore cloudtrail:DeleteResourcePolicy cloudtrail:DeleteTrail cloudtrail:DeregisterOrganizationDelegatedAdmin cloudtrail:DescribeQuery cloudtrail:DescribeTrails cloudtrail:DisableFederation cloudtrail:GenerateQuery cloudtrail:GetChannel cloudtrail:GetDashboard cloudtrail:GetEventConfiguration cloudtrail:GetEventDataStore cloudtrail:GetEventDataStoreData cloudtrail:GetEventSelectors

Prefixo do serviço	Ações do
	cloudtrail:GetImport cloudtrail:GetInsightSelectors cloudtrail:GetResourcePolicy cloudtrail:GetTrail cloudtrail:GetTrailStatus cloudtrail:ListChannels cloudtrail:ListDashboards cloudtrail:ListEventDataStores cloudtrail:ListImportFailures cloudtrail:ListImports cloudtrail:ListInsightsData cloudtrail:ListPublicKeys cloudtrail:ListQueries cloudtrail:ListTrails cloudtrail:LookupEvents cloudtrail:PutEventConfiguration cloudtrail:PutEventSelectors cloudtrail:PutInsightSelectors cloudtrail:PutResourcePolicy cloudtrail:RegisterOrganizationDelegatedAdmin cloudtrail:RestoreEventDataStore

Prefixo do serviço	Ações do
	cloudtrail:SearchSampleQueries cloudtrail:StartEventDataStoreIngestion cloudtrail:StartImport cloudtrail:StartLogging cloudtrail:StartQuery cloudtrail:StopEventDataStoreIngestion cloudtrail:StopImport cloudtrail:StopLogging cloudtrail:UpdateChannel cloudtrail:UpdateDashboard cloudtrail:UpdateEventDataStore cloudtrail:UpdateTrail

Prefixo do serviço	Ações do
cloudwatch	cloudwatch:DeleteAlarmMuteRule
	cloudwatch:DeleteAlarms
	cloudwatch:DeleteAnomalyDetector
	cloudwatch:DeleteDashboards
	cloudwatch:DeleteInsightRules
	cloudwatch:DeleteMetricStream
	cloudwatch:DescribeAlarmHistory
	cloudwatch:DescribeAlarms
	cloudwatch:DescribeAlarmsForMetric
	cloudwatch:DescribeAnomalyDetectors
	cloudwatch:DescribeInsightRules
	cloudwatch:DisableAlarmActions
	cloudwatch:DisableInsightRules
	cloudwatch:EnableAlarmActions
	cloudwatch:EnableInsightRules
	cloudwatch:GetAlarmMuteRule
	cloudwatch:GetDashboard
	cloudwatch:GetDataset
	cloudwatch:GetInsightRuleReport
	cloudwatch:GetMetricStatistics
	cloudwatch:GetMetricStream

Prefixo do serviço	Ações do
	cloudwatch:ListAlarmMuteRules
	cloudwatch:ListDashboards
	cloudwatch:ListManagedInsightRules
	cloudwatch:ListMetricStreams
	cloudwatch:PutAlarmMuteRule
	cloudwatch:PutAnomalyDetector
	cloudwatch:PutCompositeAlarm
	cloudwatch:PutDashboard
	cloudwatch:PutInsightRule
	cloudwatch:PutLogAlarm
	cloudwatch:PutManagedInsightRules
	cloudwatch:PutMetricAlarm
	cloudwatch:PutMetricStream
	cloudwatch:SetAlarmState
	cloudwatch:StartMetricStreams
	cloudwatch:StopMetricStreams

Prefixo do serviço	Ações do
codeartifact	codeartifact:AssociateExternalConnection codeartifact:CopyPackageVersions codeartifact:CreateDomain codeartifact:CreateRepository codeartifact:DeleteDomain codeartifact:DeleteDomainPermissionsPolicy codeartifact:DeletePackage codeartifact:DeletePackageVersions codeartifact:DeleteRepository codeartifact:DeleteRepositoryPermissionsPolicy codeartifact:DescribeDomain codeartifact:DescribePackage codeartifact:DescribePackageVersion codeartifact:DescribeRepository codeartifact:DisassociateExternalConnection codeartifact:DisposePackageVersions codeartifact:GetAssociatedPackageGroup codeartifact:GetAuthorizationToken codeartifact:GetDomainPermissionsPolicy codeartifact:GetPackageVersionAsset codeartifact:GetPackageVersionReadme

Prefixo do serviço	Ações do
	codeartifact:GetRepositoryEndpoint codeartifact:GetRepositoryPermissionsPolicy codeartifact:ListDomains codeartifact:ListPackageGroups codeartifact:ListPackageVersionAssets codeartifact:ListPackageVersionDependencies codeartifact:ListPackageVersions codeartifact:ListPackages codeartifact:ListRepositories codeartifact:ListRepositoriesInDomain codeartifact:PublishPackageVersion codeartifact:PutDomainPermissionsPolicy codeartifact:PutPackageMetadata codeartifact:PutPackageOriginConfiguration codeartifact:PutRepositoryPermissionsPolicy codeartifact:ReadFromRepository codeartifact:UpdatePackageVersionsStatus codeartifact:UpdateRepository

Prefixo do serviço	Ações do
codedeploy	codedeploy:BatchGetApplicationRevisions
	codedeploy:BatchGetApplications
	codedeploy:BatchGetDeploymentGroups
	codedeploy:BatchGetDeploymentInstances
	codedeploy:BatchGetDeploymentTargets
	codedeploy:BatchGetDeployments
	codedeploy:BatchGetOnPremisesInstances
	codedeploy:ContinueDeployment
	codedeploy:CreateApplication
	codedeploy:CreateDeployment
	codedeploy:CreateDeploymentConfig
	codedeploy:CreateDeploymentGroup
	codedeploy:DeleteApplication
	codedeploy:DeleteDeploymentConfig
	codedeploy:DeleteDeploymentGroup
	codedeploy:DeleteGitHubAccountToken
	codedeploy:DeleteResourcesByExternalId
	codedeploy:DeregisterOnPremisesInstance
	codedeploy:GetApplication
	codedeploy:GetApplicationRevision
	codedeploy:GetDeployment

Prefixo do serviço	Ações do
	codedeploy:GetDeploymentConfig
	codedeploy:GetDeploymentGroup
	codedeploy:GetDeploymentInstance
	codedeploy:GetDeploymentTarget
	codedeploy:GetOnPremisesInstance
	codedeploy:ListApplicationRevisions
	codedeploy:ListApplications
	codedeploy:ListDeploymentConfigs
	codedeploy:ListDeploymentGroups
	codedeploy:ListDeploymentInstances
	codedeploy:ListDeploymentTargets
	codedeploy:ListDeployments
	codedeploy:ListGitHubAccountTokenNames
	codedeploy:ListOnPremisesInstances
	codedeploy:PutLifecycleEventHookExecutionStatus
	codedeploy:RegisterApplicationRevision
	codedeploy:RegisterOnPremisesInstance
	codedeploy:SkipWaitTimeForInstanceTermination
	codedeploy:StopDeployment
	codedeploy:UpdateApplication
	codedeploy:UpdateDeploymentGroup

Prefixo do serviço	Ações do
codeguru-profiler	codeguru-profiler:AddNotificationChannels codeguru-profiler:BatchGetFrameMetricData codeguru-profiler:CreateProfilingGroup codeguru-profiler:DeleteProfilingGroup codeguru-profiler:DescribeProfilingGroup codeguru-profiler:GetFindingsReportAccountSummary codeguru-profiler:GetNotificationConfiguration codeguru-profiler:GetPolicy codeguru-profiler:GetProfile codeguru-profiler:GetRecommendations codeguru-profiler:ListFindingsReports codeguru-profiler:ListProfileTimes codeguru-profiler:ListProfilingGroups codeguru-profiler:PutPermission codeguru-profiler:RemoveNotificationChannel codeguru-profiler:RemovePermission codeguru-profiler:SubmitFeedback codeguru-profiler:UpdateProfilingGroup

Prefixo do serviço	Ações do
codeguru-reviewer	codeguru-reviewer:AssociateRepository
	codeguru-reviewer:CreateCodeReview
	codeguru-reviewer:DescribeCodeReview
	codeguru-reviewer:DescribeRecommendationFeedback
	codeguru-reviewer:DescribeRepositoryAssociation
	codeguru-reviewer:DisassociateRepository
	codeguru-reviewer:ListCodeReviews
	codeguru-reviewer:ListRecommendationFeedback
	codeguru-reviewer:ListRecommendations
	codeguru-reviewer:ListRepositoryAssociations
	codeguru-reviewer:PutRecommendationFeedback

Prefixo do serviço	Ações do
codepipeline	codepipeline:AcknowledgeJob codepipeline:AcknowledgeThirdPartyJob codepipeline:CreateCustomActionType codepipeline:CreatePipeline codepipeline:DeleteCustomActionType codepipeline:DeletePipeline codepipeline:DeleteWebhook codepipeline:DeregisterWebhookWithThirdParty codepipeline:GetActionType codepipeline:GetJobDetails codepipeline:GetPipeline codepipeline:GetPipelineExecution codepipeline:GetPipelineState codepipeline:GetThirdPartyJobDetails codepipeline:ListActionExecutions codepipeline:ListActionTypes codepipeline:ListPipelineExecutions codepipeline:ListPipelines codepipeline:ListRuleExecutions codepipeline:ListRuleTypes codepipeline:ListWebhooks

Prefixo do serviço	Ações do
	codepipeline:OverrideStageCondition codepipeline:PollForJobs codepipeline:PollForThirdPartyJobs codepipeline:PutActionRevision codepipeline:PutApprovalResult codepipeline:PutJobFailureResult codepipeline:PutJobSuccessResult codepipeline:PutThirdPartyJobFailureResult codepipeline:PutThirdPartyJobSuccessResult codepipeline:PutWebhook codepipeline:RegisterWebhookWithThirdParty codepipeline:RollbackStage codepipeline:StartPipelineExecution codepipeline:StopPipelineExecution codepipeline:UpdateActionType codepipeline:UpdatePipeline

Prefixo do serviço	Ações do
codestar	codestar:AssociateTeamMember
	codestar:CreateProject
	codestar:CreateUserProfile
	codestar:DeleteProject
	codestar:DeleteUserProfile
	codestar:DescribeProject
	codestar:DescribeUserProfile
	codestar:DisassociateTeamMember
	codestar:ListProjects
	codestar:ListResources
	codestar:ListTeamMembers
	codestar:ListUserProfiles
	codestar:UpdateProject
	codestar:UpdateTeamMember
	codestar:UpdateUserProfile

Prefixo do serviço	Ações do
codestar-notifications	codestar-notifications:CreateNotificationRule codestar-notifications:DeleteNotificationRule codestar-notifications:DeleteTarget codestar-notifications:DescribeNotificationRule codestar-notifications:ListEventTypes codestar-notifications:ListNotificationRules codestar-notifications:ListTargets codestar-notifications:Subscribe codestar-notifications:Unsubscribe codestar-notifications:UpdateNotificationRule

Prefixo do serviço	Ações do
cognito-identity	cognito-identity:CreateIdentityPool
	cognito-identity:DeleteIdentities
	cognito-identity:DeleteIdentityPool
	cognito-identity:DescribeIdentity
	cognito-identity:DescribeIdentityPool
	cognito-identity:GetIdentityPoolRoles
	cognito-identity:ListIdentities
	cognito-identity:ListIdentityPools
	cognito-identity:LookupDeveloperIdentity
	cognito-identity:MergeDeveloperIdentities
	cognito-identity:SetIdentityPoolRoles
	cognito-identity:UnlinkDeveloperIdentity
	cognito-identity:UpdateIdentityPool

Prefixo do serviço	Ações do
cognito-idp	cognito-idp:AddCustomAttributes cognito-idp:AddUserPoolClientSecret cognito-idp:AdminAddUserToGroup cognito-idp:AdminConfirmSignUp cognito-idp:AdminCreateUser cognito-idp:AdminDeleteUser cognito-idp:AdminDeleteUserAttributes cognito-idp:AdminDisableProviderForUser cognito-idp:AdminDisableUser cognito-idp:AdminEnableUser cognito-idp:AdminForgetDevice cognito-idp:AdminGetDevice cognito-idp:AdminGetUser cognito-idp:AdminGetUserAuthFactors cognito-idp:AdminInitiateAuth cognito-idp:AdminLinkProviderForUser cognito-idp:AdminListDevices cognito-idp:AdminListGroupsForUser cognito-idp:AdminListUserAuthEvents cognito-idp:AdminRemoveUserFromGroup cognito-idp:AdminResetUserPassword

Prefixo do serviço	Ações do
	cognito-idp:AdminRespondToAuthChallenge cognito-idp:AdminSetUserMFAPreference cognito-idp:AdminSetUserPassword cognito-idp:AdminSetUserSettings cognito-idp:AdminUpdateAuthEventFeedback cognito-idp:AdminUpdateDeviceStatus cognito-idp:AdminUpdateUserAttributes cognito-idp:AdminUserGlobalSignOut cognito-idp:AssociateSoftwareToken cognito-idp:ChangePassword cognito-idp:ConfirmDevice cognito-idp:ConfirmForgotPassword cognito-idp:ConfirmSignUp cognito-idp:CreateGroup cognito-idp:CreateIdentityProvider cognito-idp:CreateManagedLoginBranding cognito-idp:CreateResourceServer cognito-idp:CreateTerms cognito-idp:CreateUserImportJob cognito-idp:CreateUserPool cognito-idp:CreateUserPoolClient

Prefixo do serviço	Ações do
	cognito-idp:CreateUserPoolDomain cognito-idp:CreateUserPoolReplica cognito-idp:DeleteGroup cognito-idp:DeleteIdentityProvider cognito-idp:DeleteManagedLoginBranding cognito-idp:DeleteResourceServer cognito-idp:DeleteTerms cognito-idp:DeleteUser cognito-idp:DeleteUserAttributes cognito-idp:DeleteUserPool cognito-idp:DeleteUserPoolClient cognito-idp:DeleteUserPoolClientSecret cognito-idp:DeleteUserPoolDomain cognito-idp:DeleteUserPoolReplica cognito-idp:DescribeIdentityProvider cognito-idp:DescribeManagedLoginBranding cognito-idp:DescribeManagedLoginBrandingByClient cognito-idp:DescribeResourceServer cognito-idp:DescribeRiskConfiguration cognito-idp:DescribeTerms cognito-idp:DescribeUserImportJob

Prefixo do serviço	Ações do
	cognito-idp:DescribeUserPool cognito-idp:DescribeUserPoolClient cognito-idp:DescribeUserPoolDomain cognito-idp:ForgetDevice cognito-idp:ForgotPassword cognito-idp:GetCSVHeader cognito-idp:GetDevice cognito-idp:GetGroup cognito-idp:GetIdentityProviderByIdentifier cognito-idp:GetLogDeliveryConfiguration cognito-idp:GetProvisionedLimit cognito-idp:GetSigningCertificate cognito-idp:GetUICustomization cognito-idp:GetUser cognito-idp:GetUserAttributeVerificationCode cognito-idp:GetUserPoolMfaConfig cognito-idp:GlobalSignOut cognito-idp:InitiateAuth cognito-idp:ListDevices cognito-idp:ListGroups cognito-idp:ListIdentityProviders

Prefixo do serviço	Ações do
	cognito-idp:ListResourceServers cognito-idp:ListTerms cognito-idp:ListUserImportJobs cognito-idp:ListUserPoolClientSecrets cognito-idp:ListUserPoolClients cognito-idp:ListUserPoolReplicas cognito-idp:ListUserPools cognito-idp:ListUsers cognito-idp:ListUsersInGroup cognito-idp:ResendConfirmationCode cognito-idp:RespondToAuthChallenge cognito-idp:RevokeToken cognito-idp:SetLogDeliveryConfiguration cognito-idp:SetRiskConfiguration cognito-idp:SetUICustomization cognito-idp:SetUserMFAPreference cognito-idp:SetUserPoolMfaConfig cognito-idp:SetUserSettings cognito-idp:SignUp cognito-idp:StartUserImportJob cognito-idp:StopUserImportJob

Prefixo do serviço	Ações do
	cognito-idp:UpdateAuthEventFeedback cognito-idp:UpdateDeviceStatus cognito-idp:UpdateGroup cognito-idp:UpdateIdentityProvider cognito-idp:UpdateProvisionedLimit cognito-idp:UpdateResourceServer cognito-idp:UpdateTerms cognito-idp:UpdateUserAttributes cognito-idp:UpdateUserPool cognito-idp:UpdateUserPoolClient cognito-idp:UpdateUserPoolDomain cognito-idp:UpdateUserPoolReplica cognito-idp:VerifySoftwareToken cognito-idp:VerifyUserAttribute

Prefixo do serviço	Ações do
cognito-sync	cognito-sync:BulkPublish
	cognito-sync:DeleteDataset
	cognito-sync:DescribeDataset
	cognito-sync:DescribeIdentityPoolUsage
	cognito-sync:DescribeIdentityUsage
	cognito-sync:GetBulkPublishDetails
	cognito-sync:GetCognitoEvents
	cognito-sync:GetIdentityPoolConfiguration
	cognito-sync:ListDatasets
	cognito-sync:ListIdentityPoolUsage
	cognito-sync:ListRecords
	cognito-sync:RegisterDevice
	cognito-sync:SetCognitoEvents
	cognito-sync:SetIdentityPoolConfiguration
	cognito-sync:SubscribeToDataset
	cognito-sync:UnsubscribeFromDataset
	cognito-sync:UpdateRecords

Prefixo do serviço	Ações do
comprehendmedical	comprehendmedical:DescribeEntitiesDetectionV2Job comprehendmedical:DescribeICD10CMInferenceJob comprehendmedical:DescribePHIDetectionJob comprehendmedical:DescribeRxNormInferenceJob comprehendmedical:DescribeSNOMEDCTInferenceJob comprehendmedical:DetectEntitiesV2 comprehendmedical:DetectPHI comprehendmedical:InferICD10CM comprehendmedical:InferRxNorm comprehendmedical:InferSNOMEDCT comprehendmedical:ListEntitiesDetectionV2Jobs comprehendmedical:ListICD10CMInferenceJobs comprehendmedical:ListPHIDetectionJobs comprehendmedical:ListRxNormInferenceJobs comprehendmedical:ListSNOMEDCTInferenceJobs comprehendmedical:StartEntitiesDetectionV2Job comprehendmedical:StartICD10CMInferenceJob comprehendmedical:StartPHIDetectionJob comprehendmedical:StartRxNormInferenceJob comprehendmedical:StartSNOMEDCTInferenceJob comprehendmedical:StopEntitiesDetectionV2Job

Prefixo do serviço	Ações do
	comprehendmedical:StopICD10CMInferenceJob comprehendmedical:StopPHIDetectionJob comprehendmedical:StopRxNormInferenceJob comprehendmedical:StopSNOMEDCTInferenceJob

Prefixo do serviço	Ações do
compute-optimizer	compute-optimizer:DeleteRecommendationPreferences
	compute-optimizer:DescribeRecommendationExportJobs
	compute-optimizer:ExportAutoScalingGroupRecommendations
	compute-optimizer:ExportEBSVolumeRecommendations
	compute-optimizer:ExportEC2InstanceRecommendations
	compute-optimizer:ExportECSServiceRecommendations
	compute-optimizer:ExportIdleRecommendations
	compute-optimizer:ExportLambdaFunctionRecommendations
	compute-optimizer:ExportLicenseRecommendations
	compute-optimizer:ExportRDSDatabaseRecommendations
	compute-optimizer:GetEC2RecommendationProjectedMetrics
	compute-optimizer:GetECSServiceRecommendationProjectedMetrics
	compute-optimizer:GetEffectiveRecommendationPreferences
	compute-optimizer:GetEnrollmentStatus
	compute-optimizer:GetEnrollmentStatusesForOrganization
	compute-optimizer:GetRDSDatabaseRecommendationProjectedMetrics
	compute-optimizer:GetRecommendationPreferences
	compute-optimizer:GetRecommendationSummaries
	compute-optimizer:PutRecommendationPreferences

Prefixo do serviço	Ações do
	compute-optimizer:UpdateEnrollmentStatus

Prefixo do serviço	Ações do
config	config:BatchGetResourceConfig config:DeleteAggregationAuthorization config:DeleteConfigRule config:DeleteConfigurationAggregator config:DeleteConfigurationRecorder config:DeleteConformancePack config:DeleteDeliveryChannel config:DeleteEvaluationResults config:DeleteOrganizationConfigRule config:DeleteOrganizationConformancePack config:DeletePendingAggregationRequest config:DeleteRemediationConfiguration config:DeleteRemediationExceptions config:DeleteResourceConfig config:DeleteRetentionConfiguration config:DeleteStoredQuery config:DeliverConfigSnapshot config:DescribeAggregateComplianceByConfigRules config:DescribeAggregateComplianceByConformancePacks config:DescribeAggregationAuthorizations config:DescribeComplianceByConfigRule

Prefixo do serviço	Ações do
	config:DescribeComplianceByResource config:DescribeConfigRuleEvaluationStatus config:DescribeConfigRules config:DescribeConfigurationAggregatorSourcesStatus config:DescribeConfigurationAggregators config:DescribeConfigurationRecorderStatus config:DescribeConfigurationRecorders config:DescribeConformancePackCompliance config:DescribeConformancePackStatus config:DescribeConformancePacks config:DescribeDeliveryChannelStatus config:DescribeDeliveryChannels config:DescribeOrganizationConfigRuleStatuses config:DescribeOrganizationConfigRules config:DescribeOrganizationConformancePackStatuses config:DescribeOrganizationConformancePacks config:DescribePendingAggregationRequests config:DescribeRemediationConfigurations config:DescribeRemediationExceptions config:DescribeRemediationExecutionStatus config:DescribeRetentionConfigurations

Prefixo do serviço	Ações do
	config:GetComplianceDetailsByConfigRule config:GetComplianceDetailsByResource config:GetComplianceSummaryByConfigRule config:GetComplianceSummaryByResourceType config:GetConformancePackComplianceDetails config:GetConformancePackComplianceSummary config:GetCustomRulePolicy config:GetDiscoveredResourceCounts config:GetOrganizationConfigRuleDetailedStatus config:GetOrganizationConformancePackDetailedStatus config:GetOrganizationCustomRulePolicy config:GetResourceConfigHistory config:GetResourceEvaluationSummary config:GetStoredQuery config:ListConfigurationRecorders config:ListConformancePackComplianceScores config:ListConnectors config:ListDiscoveredResources config:ListResourceEvaluations config:ListStoredQueries config:PutConfigRule

Prefixo do serviço	Ações do
	config:PutConfigurationAggregator config:PutConfigurationRecorder config:PutConformancePack config:PutConnector config:PutDeliveryChannel config:PutEvaluations config:PutExternalEvaluation config:PutOrganizationConfigRule config:PutOrganizationConformancePack config:PutRemediationConfigurations config:PutRemediationExceptions config:PutResourceConfig config:PutRetentionConfiguration config:PutStoredQuery config:SelectResourceConfig config:StartConfigRulesEvaluation config:StartConfigurationRecorder config:StartRemediationExecution config:StartResourceEvaluation config:StopConfigurationRecorder

Prefixo do serviço	Ações do
conectar	connect:ActivateEvaluationForm
	connect:AssociateAnalyticsDataSet
	connect:AssociateApprovedOrigin
	connect:AssociateBot
	connect:AssociateContactWithUser
	connect:AssociateDefaultVocabulary
	connect:AssociateEmailAddressAlias
	connect:AssociateFlow
	connect:AssociateInstanceStorageConfig
	connect:AssociateLambdaFunction
	connect:AssociateLexBot
	connect:AssociatePhoneNumberContactFlow
	connect:AssociateQueueEmailAddresses
	connect:AssociateQueueQuickConnects
	connect:AssociateRoutingProfileQueues
	connect:AssociateSecurityKey
	connect:AssociateUserProficiencies
	connect:BatchAssociateAnalyticsDataSet
	connect:BatchCreateDataTableValue
	connect:BatchDeleteDataTableValue
	connect:BatchDescribeDataTableValue

Prefixo do serviço	Ações do
	connect:BatchDisassociateAnalyticsDataSet
	connect:BatchGetFlowAssociation
	connect:BatchPutContact
	connect:BatchUpdateDataTableValue
	connect:ClaimPhoneNumber
	connect:CreateAgentStatus
	connect:CreateAttachedFile
	connect:CreateContact
	connect:CreateContactFlow
	connect:CreateContactFlowModule
	connect:CreateContactFlowModuleAlias
	connect:CreateContactFlowModuleVersion
	connect:CreateContactFlowVersion
	connect:CreateDataTable
	connect:CreateDataTableAttribute
	connect:CreateEmailAddress
	connect:CreateEvaluationForm
	connect:CreateHoursOfOperation
	connect:CreateInstance
	connect:CreateIntegrationAssociation
	connect:CreateParticipant

Prefixo do serviço	Ações do
	connect:CreatePersistentContactAssociation
	connect:CreatePredefinedAttribute
	connect:CreatePrompt
	connect:CreatePushNotificationRegistration
	connect:CreateQueue
	connect:CreateQuickConnect
	connect:CreateRoutingProfile
	connect:CreateRule
	connect:CreateSecurityProfile
	connect:CreateTaskTemplate
	connect:CreateTrafficDistributionGroup
	connect:CreateUseCase
	connect:CreateUser
	connect:CreateUserHierarchyGroup
	connect:CreateView
	connect:CreateViewVersion
	connect:CreateVocabulary
	connect:CreateWorkspace
	connect:DeactivateEvaluationForm
	connect:DeleteContactEvaluation
	connect:DeleteContactFlow

Prefixo do serviço	Ações do
	connect:DeleteContactFlowModule
	connect:DeleteContactFlowModuleAlias
	connect:DeleteContactFlowModuleVersion
	connect:DeleteContactFlowVersion
	connect:DeleteDataTable
	connect:DeleteDataTableAttribute
	connect:DeleteEmailAddress
	connect:DeleteEvaluationForm
	connect:DeleteHoursOfOperation
	connect:DeleteHoursOfOperationOverride
	connect:DeleteInstance
	connect:DeleteIntegrationAssociation
	connect:DeleteNotification
	connect:DeletePredefinedAttribute
	connect:DeletePrompt
	connect:DeletePushNotificationRegistration
	connect:DeleteQueue
	connect:DeleteQuickConnect
	connect:DeleteRoutingProfile
	connect:DeleteRule
	connect:DeleteSecurityProfile

Prefixo do serviço	Ações do
	connect:DeleteTaskTemplate
	connect:DeleteTrafficDistributionGroup
	connect:DeleteUseCase
	connect:DeleteUser
	connect:DeleteUserHierarchyGroup
	connect:DeleteView
	connect:DeleteVocabulary
	connect:DeleteWorkspace
	connect:DeleteWorkspaceMedia
	connect:DescribeAttachedFilesConfiguration
	connect:DescribeAuthenticationProfile
	connect:DescribeContactFlowModuleAlias
	connect:DescribeDataTableAttribute
	connect:DescribeHoursOfOperationOverride
	connect:DescribeInstanceAttribute
	connect:DescribeInstanceStorageConfig
	connect:DescribeNotification
	connect:DescribePhoneNumber
	connect:DescribeRule
	connect:DescribeTrafficDistributionGroup
	connect:DescribeUserHierarchyStructure

Prefixo do serviço	Ações do
	connect:DescribeVocabulary
	connect:DisassociateAnalyticsDataSet
	connect:DisassociateApprovedOrigin
	connect:DisassociateBot
	connect:DisassociateEmailAddressAlias
	connect:DisassociateFlow
	connect:DisassociateInstanceStorageConfig
	connect:DisassociateLambdaFunction
	connect:DisassociateLexBot
	connect:DisassociatePhoneNumberContactFlow
	connect:DisassociateQueueEmailAddresses
	connect:DisassociateQueueQuickConnects
	connect:DisassociateRoutingProfileQueues
	connect:DisassociateSecurityKey
	connect:DisassociateUserProficiencies
	connect:DismissUserContact
	connect:EvaluateDataTableValues
	connect:GetContactAttributes
	connect:GetContactMetrics
	connect:GetCurrentMetricData
	connect:GetCurrentUserData

Prefixo do serviço	Ações do
	connect:GetEffectiveHoursOfOperations
	connect:GetEvaluationFormValidation
	connect:GetFederationToken
	connect:GetFlowAssociation
	connect:GetMetricData
	connect:GetMetricDataV2
	connect:GetPromptFile
	connect:GetTaskTemplate
	connect:GetTrafficDistribution
	connect:ImportPhoneNumber
	connect:ImportWorkspaceMedia
	connect:ListAnalyticsDataAssociations
	connect:ListAnalyticsDataLakeDataSets
	connect:ListApprovedOrigins
	connect:ListAssociatedContacts
	connect:ListAttachedFilesConfigurations
	connect:ListAuthenticationProfiles
	connect:ListBots
	connect:ListContactEvaluations
	connect:ListContactFlowModuleAliases
	connect:ListContactFlowModuleVersions

Prefixo do serviço	Ações do
	connect:ListContactFlowModules
	connect:ListContactFlowVersions
	connect:ListContactFlows
	connect:ListContactReferences
	connect:ListDataTableAttributes
	connect:ListDataTablePrimaryValues
	connect:ListDataTableValues
	connect:ListDataTables
	connect:ListDefaultVocabularies
	connect:ListEvaluationFormVersions
	connect:ListEvaluationForms
	connect:ListFlowAssociations
	connect:ListHoursOfOperations
	connect:ListInstanceAttributes
	connect:ListInstanceStorageConfigs
	connect:ListIntegrationAssociations
	connect:ListLambdaFunctions
	connect:ListLexBots
	connect:ListNotifications
	connect:ListPhoneNumbers
	connect:ListPhoneNumbersV2

Prefixo do serviço	Ações do
	connect:ListPredefinedAttributes
	connect:ListPrompts
	connect:ListQueueEmailAddresses
	connect:ListQueueQuickConnects
	connect:ListQueues
	connect:ListQuickConnects
	connect:ListRealtimeContactAnalysisSegmentsV2
	connect:ListRoutingProfileManualAssignmentQueues
	connect:ListRoutingProfileQueues
	connect:ListRoutingProfiles
	connect:ListRules
	connect:ListSecurityKeys
	connect:ListSecurityProfileApplications
	connect:ListSecurityProfileFlowModules
	connect:ListSecurityProfilePermissions
	connect:ListSecurityProfiles
	connect:ListTaskTemplates
	connect:ListTrafficDistributionGroups
	connect:ListUseCases
	connect:ListUserHierarchyGroups
	connect:ListUserNotifications

Prefixo do serviço	Ações do
	connect:ListUsers
	connect:ListViewVersions
	connect:ListViews
	connect:ListWorkspaceMedia
	connect:ListWorkspacePages
	connect:ListWorkspaces
	connect:MonitorContact
	connect:PauseContact
	connect:PutUserStatus
	connect:ReleasePhoneNumber
	connect:ReplicateInstance
	connect:ResumeContact
	connect:ResumeContactRecording
	connect:SearchAgentStatuses
	connect:SearchAvailablePhoneNumbers
	connect:SearchContactEvaluations
	connect:SearchContactFlowModules
	connect:SearchContactFlows
	connect:SearchContacts
	connect:SearchDataTables
	connect:SearchEmailAddresses

Prefixo do serviço	Ações do
	connect:SearchEvaluationForms
	connect:SearchHoursOfOperations
	connect:SearchPredefinedAttributes
	connect:SearchPrompts
	connect:SearchQueues
	connect:SearchQuickConnects
	connect:SearchRoutingProfiles
	connect:SearchRules
	connect:SearchSecurityProfiles
	connect:SearchUserHierarchyGroups
	connect:SearchViews
	connect:SearchVocabularies
	connect:SearchWorkspaceAssociations
	connect:SearchWorkspaces
	connect:SendChatIntegrationEvent
	connect:SendOutboundEmail
	connect:StartChatContact
	connect:StartContactConversationalAnalyticsJob
	connect:StartContactEvaluation
	connect:StartContactMediaProcessing
	connect:StartContactRecording

Prefixo do serviço	Ações do
	connect:StartContactStreaming
	connect:StartEmailContact
	connect:StartOutboundChatContact
	connect:StartOutboundEmailContact
	connect:StartOutboundVoiceContact
	connect:StartScreenSharing
	connect:StartTaskContact
	connect:StartWebRTCContact
	connect:StopContact
	connect:StopContactMediaProcessing
	connect:StopContactRecording
	connect:StopContactStreaming
	connect:SubmitContactEvaluation
	connect:SuspendContactRecording
	connect:TransferContact
	connect:UpdateAgentStatus
	connect:UpdateAttachedFilesConfiguration
	connect:UpdateAuthenticationProfile
	connect:UpdateContact
	connect:UpdateContactAttributes
	connect:UpdateContactEvaluation

Prefixo do serviço	Ações do
	connect:UpdateContactFlowContent
	connect:UpdateContactFlowMetadata
	connect:UpdateContactFlowModuleAlias
	connect:UpdateContactFlowModuleContent
	connect:UpdateContactFlowModuleMetadata
	connect:UpdateContactFlowName
	connect:UpdateContactRoutingData
	connect:UpdateContactSchedule
	connect:UpdateDataTableAttribute
	connect:UpdateDataTableMetadata
	connect:UpdateDataTablePrimaryValues
	connect:UpdateEmailAddressMetadata
	connect:UpdateEvaluationForm
	connect:UpdateHoursOfOperation
	connect:UpdateHoursOfOperationOverride
	connect:UpdateInstanceAttribute
	connect:UpdateInstanceStorageConfig
	connect:UpdateNotificationContent
	connect:UpdateParticipantAuthentication
	connect:UpdateParticipantRoleConfig
	connect:UpdatePhoneNumber

Prefixo do serviço	Ações do
	connect:UpdatePhoneNumberMetadata
	connect:UpdatePredefinedAttribute
	connect:UpdatePrompt
	connect:UpdateQueueHoursOfOperation
	connect:UpdateQueueMaxContacts
	connect:UpdateQueueName
	connect:UpdateQueueOutboundCallerConfig
	connect:UpdateQueueOutboundEmailConfig
	connect:UpdateQueueStatus
	connect:UpdateQuickConnectConfig
	connect:UpdateQuickConnectName
	connect:UpdateRoutingProfileAgentAvailabilityTimer
	connect:UpdateRoutingProfileConcurrency
	connect:UpdateRoutingProfileDefaultOutboundQueue
	connect:UpdateRoutingProfileName
	connect:UpdateRoutingProfileQueues
	connect:UpdateRule
	connect:UpdateSecurityProfile
	connect:UpdateTaskTemplate
	connect:UpdateTrafficDistribution
	connect:UpdateUserHierarchy

Prefixo do serviço	Ações do
	connect:UpdateUserHierarchyGroupName connect:UpdateUserHierarchyStructure connect:UpdateUserIdentityInfo connect:UpdateUserNotificationStatus connect:UpdateUserPhoneConfig connect:UpdateUserProficiencies connect:UpdateUserRoutingProfile connect:UpdateUserSecurityProfiles connect:UpdateViewContent connect:UpdateViewMetadata connect:UpdateWorkspaceMetadata connect:UpdateWorkspaceTheme connect:UpdateWorkspaceVisibility
cur	cur:DeleteReportDefinition cur:DescribeReportDefinitions cur:ModifyReportDefinition cur:PutReportDefinition

Prefixo do serviço	Ações do
databrew	databrew:BatchDeleteRecipeVersion
	databrew:CreateDataset
	databrew:CreateProfileJob
	databrew:CreateProject
	databrew:CreateRecipe
	databrew:CreateRecipeJob
	databrew:CreateRuleset
	databrew:CreateSchedule
	databrew:DeleteDataset
	databrew:DeleteJob
	databrew:DeleteProject
	databrew:DeleteRecipeVersion
	databrew:DeleteRuleset
	databrew:DeleteSchedule
	databrew:DescribeDataset
	databrew:DescribeJob
	databrew:DescribeJobRun
	databrew:DescribeProject
	databrew:DescribeRecipe
	databrew:DescribeRuleset
	databrew:DescribeSchedule

Prefixo do serviço	Ações do
	databrew:ListDatasets
	databrew:ListJobRuns
	databrew:ListJobs
	databrew:ListProjects
	databrew:ListRecipeVersions
	databrew:ListRecipes
	databrew:ListRulesets
	databrew:ListSchedules
	databrew:PublishRecipe
	databrew:SendProjectSessionAction
	databrew:StartJobRun
	databrew:StartProjectSession
	databrew:StopJobRun
	databrew:UpdateDataset
	databrew:UpdateProfileJob
	databrew:UpdateProject
	databrew:UpdateRecipe
	databrew:UpdateRecipeJob
	databrew:UpdateRuleset
	databrew:UpdateSchedule

Prefixo do serviço	Ações do
dataexchange	dataexchange:AcceptDataGrant dataexchange:CancelJob dataexchange:CreateDataGrant dataexchange:CreateDataSet dataexchange:CreateEventAction dataexchange:CreateJob dataexchange:CreateRevision dataexchange:DeleteAsset dataexchange:DeleteDataGrant dataexchange:DeleteEventAction dataexchange:DeleteRevision dataexchange:GetDataGrant dataexchange:GetEventAction dataexchange:GetJob dataexchange:GetReceivedDataGrant dataexchange:ListDataGrants dataexchange:ListDataSetRevisions dataexchange:ListDataSets dataexchange:ListEventActions dataexchange:ListJobs dataexchange:ListReceivedDataGrants

Prefixo do serviço	Ações do
	dataexchange:ListRevisionAssets dataexchange:RevokeRevision dataexchange:SendDataSetNotification dataexchange:StartJob dataexchange:UpdateAsset dataexchange:UpdateDataSet dataexchange:UpdateEventAction dataexchange:UpdateRevision

Prefixo do serviço	Ações do
datapipeline	datapipeline:ActivatePipeline datapipeline:CreatePipeline datapipeline:DeactivatePipeline datapipeline:DeletePipeline datapipeline:DescribeObjects datapipeline:DescribePipelines datapipeline:EvaluateExpression datapipeline:GetPipelineDefinition datapipeline:ListPipelines datapipeline:PollForTask datapipeline:PutPipelineDefinition datapipeline:QueryObjects datapipeline:ReportTaskProgress datapipeline:ReportTaskRunnerHeartbeat datapipeline:SetStatus datapipeline:SetTaskStatus datapipeline:ValidatePipelineDefinition

Prefixo do serviço	Ações do
dax	dax:CreateCluster dax:DecreaseReplicationFactor dax:DeleteCluster dax:DeleteParameterGroup dax:DeleteSubnetGroup dax:DescribeClusters dax:DescribeDefaultParameters dax:DescribeEvents dax:DescribeParameterGroups dax:DescribeParameters dax:DescribeSubnetGroups dax:IncreaseReplicationFactor dax:RebootNode dax:UpdateCluster dax:UpdateParameterGroup dax:UpdateSubnetGroup

Prefixo do serviço	Ações do
devicefarm	devicefarm:CreateDevicePool
	devicefarm:CreateInstanceProfile
	devicefarm:CreateNetworkProfile
	devicefarm:CreateProject
	devicefarm:CreateRemoteAccessSession
	devicefarm:CreateTestGridProject
	devicefarm:CreateTestGridUrl
	devicefarm:CreateUpload
	devicefarm:CreateVPCEConfiguration
	devicefarm:DeleteDevicePool
	devicefarm:DeleteInstanceProfile
	devicefarm:DeleteNetworkProfile
	devicefarm:DeleteProject
	devicefarm:DeleteRemoteAccessSession
	devicefarm:DeleteRun
	devicefarm:DeleteTestGridProject
	devicefarm:DeleteUpload
	devicefarm:DeleteVPCEConfiguration
	devicefarm:GetAccountSettings
	devicefarm:GetDevice
	devicefarm:GetDeviceInstance

Prefixo do serviço	Ações do
	devicefarm:GetDevicePool
	devicefarm:GetDevicePoolCompatibility
	devicefarm:GetInstanceProfile
	devicefarm:GetJob
	devicefarm:GetNetworkProfile
	devicefarm:GetOfferingStatus
	devicefarm:GetProject
	devicefarm:GetRemoteAccessSession
	devicefarm:GetRun
	devicefarm:GetSuite
	devicefarm:GetTest
	devicefarm:GetTestGridProject
	devicefarm:GetTestGridSession
	devicefarm:GetUpload
	devicefarm:GetVPCEConfiguration
	devicefarm:ListArtifacts
	devicefarm:ListDeviceInstances
	devicefarm:ListDevicePools
	devicefarm:ListDevices
	devicefarm:ListInstanceProfiles
	devicefarm:ListJobs

Prefixo do serviço	Ações do
	devicefarm:ListNetworkProfiles
	devicefarm:ListOfferingPromotions
	devicefarm:ListOfferingTransactions
	devicefarm:ListOfferings
	devicefarm:ListProjects
	devicefarm:ListRemoteAccessSessions
	devicefarm:ListRuns
	devicefarm:ListSamples
	devicefarm:ListSuites
	devicefarm:ListTestGridProjects
	devicefarm:ListTestGridSessionActions
	devicefarm:ListTestGridSessionArtifacts
	devicefarm:ListTestGridSessions
	devicefarm:ListTests
	devicefarm:ListUniqueProblems
	devicefarm:ListUploads
	devicefarm:ListVPCEConfigurations
	devicefarm:PurchaseOffering
	devicefarm:RenewOffering
	devicefarm:ScheduleRun
	devicefarm:StopJob

Prefixo do serviço	Ações do
	devicefarm:StopRemoteAccessSession
	devicefarm:StopRun
	devicefarm:UpdateDeviceInstance
	devicefarm:UpdateDevicePool
	devicefarm:UpdateInstanceProfile
	devicefarm:UpdateNetworkProfile
	devicefarm:UpdateProject
	devicefarm:UpdateTestGridProject
	devicefarm:UpdateUpload
	devicefarm:UpdateVPCEConfiguration

Prefixo do serviço	Ações do
devops-guru	devops-guru:AddNotificationChannel devops-guru:DeleteInsight devops-guru:DescribeAccountHealth devops-guru:DescribeAccountOverview devops-guru:DescribeAnomaly devops-guru:DescribeEventSourcesConfig devops-guru:DescribeFeedback devops-guru:DescribeInsight devops-guru:DescribeOrganizationHealth devops-guru:DescribeOrganizationOverview devops-guru:DescribeOrganizationResourceCollectionHealth devops-guru:DescribeResourceCollectionHealth devops-guru:DescribeServiceIntegration devops-guru:GetCostEstimation devops-guru:GetResourceCollection devops-guru:ListAnomaliesForInsight devops-guru:ListAnomalousLogGroups devops-guru:ListEvents devops-guru:ListInsights devops-guru:ListMonitoredResources devops-guru:ListNotificationChannels

Prefixo do serviço	Ações do
	devops-guru:ListOrganizationInsights devops-guru:ListRecommendations devops-guru:PutFeedback devops-guru:RemoveNotificationChannel devops-guru:SearchInsights devops-guru:SearchOrganizationInsights devops-guru:StartCostEstimation devops-guru:UpdateEventSourcesConfig devops-guru:UpdateResourceCollection devops-guru:UpdateServiceIntegration

Prefixo do serviço	Ações do
directconnect	directconnect:AcceptDirectConnectGatewayAssociationProposal
	directconnect:AllocateConnectionOnInterconnect
	directconnect:AllocateHostedConnection
	directconnect:AllocatePrivateVirtualInterface
	directconnect:AllocatePublicVirtualInterface
	directconnect:AllocateTransitVirtualInterface
	directconnect:AssociateConnectionWithLag
	directconnect:AssociateHostedConnection
	directconnect:AssociateMacSecKey
	directconnect:AssociateVirtualInterface
	directconnect:ConfirmConnection
	directconnect:ConfirmCustomerAgreement
	directconnect:ConfirmPrivateVirtualInterface
	directconnect:ConfirmPublicVirtualInterface
	directconnect:ConfirmTransitVirtualInterface
	directconnect:CreateBGPPeer
	directconnect:CreateConnection
	directconnect:CreateDirectConnectGateway
	directconnect:CreateDirectConnectGatewayAssociation
	directconnect:CreateDirectConnectGatewayAssociationProposal
	directconnect:CreateInterconnect

Prefixo do serviço	Ações do
	directconnect:CreateLag
	directconnect:CreatePrivateVirtualInterface
	directconnect:CreatePublicVirtualInterface
	directconnect:CreateTransitVirtualInterface
	directconnect:DeleteBGPPeer
	directconnect:DeleteConnection
	directconnect:DeleteDirectConnectGateway
	directconnect:DeleteDirectConnectGatewayAssociation
	directconnect:DeleteDirectConnectGatewayAssociationProposal
	directconnect:DeleteInterconnect
	directconnect:DeleteLag
	directconnect:DeleteVirtualInterface
	directconnect:DescribeConnectionLoa
	directconnect:DescribeConnections
	directconnect:DescribeConnectionsOnInterconnect
	directconnect:DescribeCustomerMetadata
	directconnect:DescribeDirectConnectGatewayAssociationProposals
	directconnect:DescribeDirectConnectGatewayAssociations
	directconnect:DescribeDirectConnectGatewayAttachments
	directconnect:DescribeDirectConnectGateways
	directconnect:DescribeHostedConnections

Prefixo do serviço	Ações do
	directconnect:DescribeInterconnectLoa
	directconnect:DescribeInterconnects
	directconnect:DescribeLags
	directconnect:DescribeLoa
	directconnect:DescribeLocations
	directconnect:DescribeRouterConfiguration
	directconnect:DescribeVirtualGateways
	directconnect:DescribeVirtualInterfaces
	directconnect:DisassociateConnectionFromLag
	directconnect:DisassociateMacSecKey
	directconnect:ListVirtualInterfaceTestHistory
	directconnect:StartBgpFailoverTest
	directconnect:StopBgpFailoverTest
	directconnect:UpdateConnection
	directconnect:UpdateDirectConnectGateway
	directconnect:UpdateDirectConnectGatewayAssociation
	directconnect:UpdateLag
	directconnect:UpdateVirtualInterfaceAttributes

Prefixo do serviço	Ações do
dlm	dlm:CreateLifecyclePolicy dlm:DeleteLifecyclePolicy dlm:GetLifecyclePolicies dlm:GetLifecyclePolicy dlm:UpdateLifecyclePolicy

Prefixo do serviço	Ações do
dms	dms:ApplyPendingMaintenanceAction
	dms:AssociateExtensionPack
	dms:BatchStartRecommendations
	dms:CancelMetadataModelCreation
	dms:CancelReplicationTaskAssessmentRun
	dms:CreateDataProvider
	dms:CreateEndpoint
	dms:CreateEventSubscription
	dms:CreateInstanceProfile
	dms:CreateMigrationProject
	dms:CreateReplicationConfig
	dms:CreateReplicationInstance
	dms:CreateReplicationSubnetGroup
	dms:CreateReplicationTask
	dms:DeleteCertificate
	dms:DeleteConnection
	dms:DeleteDataMigration
	dms:DeleteDataProvider
	dms:DeleteEndpoint
	dms:DeleteEventSubscription
	dms:DeleteFleetAdvisorCollector

Prefixo do serviço	Ações do
	dms:DeleteFleetAdvisorDatabases
	dms:DeleteInstanceProfile
	dms:DeleteMigrationProject
	dms:DeleteReplicationConfig
	dms:DeleteReplicationInstance
	dms:DeleteReplicationSubnetGroup
	dms:DeleteReplicationTask
	dms:DeleteReplicationTaskAssessmentRun
	dms:DescribeAccountAttributes
	dms:DescribeApplicableIndividualAssessments
	dms:DescribeCertificates
	dms:DescribeConnections
	dms:DescribeDataMigrations
	dms:DescribeEndpointSettings
	dms:DescribeEndpointTypes
	dms:DescribeEndpoints
	dms:DescribeEngineVersions
	dms:DescribeEventCategories
	dms:DescribeEventSubscriptions
	dms:DescribeEvents
	dms:DescribeFleetAdvisorCollectors

Prefixo do serviço	Ações do
	dms:DescribeFleetAdvisorDatabases
	dms:DescribeFleetAdvisorLsaAnalysis
	dms:DescribeFleetAdvisorSchemaObjectSummary
	dms:DescribeFleetAdvisorSchemas
	dms:DescribeMetadataModel
	dms:DescribeMetadataModelChildren
	dms:DescribeMetadataModelCreations
	dms:DescribeMetadataModelImports
	dms:DescribeOrderableReplicationInstances
	dms:DescribePendingMaintenanceActions
	dms:DescribeRecommendationLimitations
	dms:DescribeRecommendations
	dms:DescribeRefreshSchemasStatus
	dms:DescribeReplicationConfigs
	dms:DescribeReplicationInstanceTaskLogs
	dms:DescribeReplicationInstances
	dms:DescribeReplicationSubnetGroups
	dms:DescribeReplicationTableStatistics
	dms:DescribeReplicationTaskAssessmentResults
	dms:DescribeReplicationTaskAssessmentRuns
	dms:DescribeReplicationTaskIndividualAssessments

Prefixo do serviço	Ações do
	dms:DescribeReplicationTasks
	dms:DescribeReplications
	dms:DescribeSchemas
	dms:DescribeTableStatistics
	dms:ExportMetadataModelAssessment
	dms:ImportCertificate
	dms:ListDataProviders
	dms:ListExtensionPacks
	dms:ListInstanceProfiles
	dms:ListMetadataModelAssessments
	dms:ListMetadataModelConversions
	dms:ListMetadataModelExports
	dms:ListMigrationProjects
	dms:ModifyDataMigration
	dms:ModifyEndpoint
	dms:ModifyEventSubscription
	dms:ModifyReplicationConfig
	dms:ModifyReplicationInstance
	dms:ModifyReplicationSubnetGroup
	dms:ModifyReplicationTask
	dms:MoveReplicationTask

Prefixo do serviço	Ações do
	dms:RebootReplicationInstance
	dms:RefreshSchemas
	dms:ReloadReplicationTables
	dms:ReloadTables
	dms:RunFleetAdvisorLsaAnalysis
	dms:StartMetadataModelAssessment
	dms:StartMetadataModelConversion
	dms:StartMetadataModelCreation
	dms:StartMetadataModelExportAsScripts
	dms:StartMetadataModelExportToTarget
	dms:StartRecommendations
	dms:StartReplication
	dms:StartReplicationTask
	dms:StartReplicationTaskAssessment
	dms:StopDataMigration
	dms:StopReplicationTask
	dms:TestConnection
	dms:UpdateConversionConfiguration
	dms:UpdateDataProvider
	dms:UpdateInstanceProfile
	dms:UpdateMigrationProject

Prefixo do serviço	Ações do
	dms:UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic:ApplyPendingMaintenanceAction docdb-elastic:CopyClusterSnapshot docdb-elastic:DeleteCluster docdb-elastic:DeleteClusterSnapshot docdb-elastic:GetCluster docdb-elastic:GetClusterSnapshot docdb-elastic:GetPendingMaintenanceAction docdb-elastic:ListClusterSnapshots docdb-elastic:ListClusters docdb-elastic:ListPendingMaintenanceActions docdb-elastic:RestoreClusterFromSnapshot docdb-elastic:StartCluster docdb-elastic:StopCluster docdb-elastic:UpdateCluster

Prefixo do serviço	Ações do
dynamodb	dynamodb:CreateBackup
	dynamodb:CreateGlobalTable
	dynamodb:CreateTable
	dynamodb:DeleteBackup
	dynamodb:DeleteTable
	dynamodb:DescribeBackup
	dynamodb:DescribeContinuousBackups
	dynamodb:DescribeContributorInsights
	dynamodb:DescribeEndpoints
	dynamodb:DescribeExport
	dynamodb:DescribeGlobalTable
	dynamodb:DescribeGlobalTableSettings
	dynamodb:DescribeImport
	dynamodb:DescribeKinesisStreamingDestination
	dynamodb:DescribeLimits
	dynamodb:DescribeStream
	dynamodb:DescribeTable
	dynamodb:DescribeTableReplicaAutoScaling
	dynamodb:DescribeTimeToLive
	dynamodb:DisableKinesisStreamingDestination
	dynamodb:EnableKinesisStreamingDestination

Prefixo do serviço	Ações do
	dynamodb:ExportTableToPointInTime
	dynamodb:GetResourcePolicy
	dynamodb:ImportTable
	dynamodb:ListBackups
	dynamodb:ListContributorInsights
	dynamodb:ListExports
	dynamodb:ListGlobalTables
	dynamodb:ListImports
	dynamodb:ListStreams
	dynamodb:ListTables
	dynamodb:RestoreTableFromBackup
	dynamodb:RestoreTableToPointInTime
	dynamodb:UpdateContinuousBackups
	dynamodb:UpdateContributorInsights
	dynamodb:UpdateGlobalTable
	dynamodb:UpdateGlobalTableSettings
	dynamodb:UpdateKinesisStreamingDestination
	dynamodb:UpdateTable
	dynamodb:UpdateTableReplicaAutoScaling
	dynamodb:UpdateTimeToLive

Prefixo do serviço	Ações do
ebs	ebs:CompleteSnapshot ebs:StartSnapshot

Prefixo do serviço	Ações do
ec2	ec2:AcceptAddressTransfer
	ec2:AcceptCapacityReservationBillingOwnership
	ec2:AcceptReservedInstancesExchangeQuote
	ec2:AcceptTransitGatewayClientVpnAttachment
	ec2:AcceptTransitGatewayMulticastDomainAssociations
	ec2:AcceptTransitGatewayPeeringAttachment
	ec2:AcceptTransitGatewayVpcAttachment
	ec2:AcceptVpcEndpointConnections
	ec2:AcceptVpcPeeringConnection
	ec2:AdvertiseByoipCidr
	ec2:AllocateAddress
	ec2:AllocateHosts
	ec2:AllocateIpamPoolCidr
	ec2:ApplySecurityGroupsToClientVpnTargetNetwork
	ec2:AssignIpv6Addresses
	ec2:AssignPrivateIpAddresses
	ec2:AssignPrivateNatGatewayAddress
	ec2:AssociateAddress
	ec2:AssociateCapacityReservationBillingOwner
	ec2:AssociateClientVpnTargetNetwork
	ec2:AssociateDhcpOptions

Prefixo do serviço	Ações do
	ec2:AssociateEnclaveCertificateIamRole
	ec2:AssociateIamInstanceProfile
	ec2:AssociateInstanceEventWindow
	ec2:AssociateIpamByoasn
	ec2:AssociateIpamResourceDiscovery
	ec2:AssociateNatGatewayAddress
	ec2:AssociateRouteServer
	ec2:AssociateRouteTable
	ec2:AssociateSecurityGroupVpc
	ec2:AssociateSubnetCidrBlock
	ec2:AssociateTransitGatewayMulticastDomain
	ec2:AssociateTransitGatewayPolicyTable
	ec2:AssociateTransitGatewayRouteTable
	ec2:AssociateTrunkInterface
	ec2:AssociateVpcCidrBlock
	ec2:AttachClassicLinkVpc
	ec2:AttachImageWatermark
	ec2:AttachInternetGateway
	ec2:AttachNetworkInterface
	ec2:AttachVerifiedAccessTrustProvider
	ec2:AttachVolume

Prefixo do serviço	Ações do
	ec2:AttachVpnGateway
	ec2:AuthorizeClientVpnIngress
	ec2:AuthorizeSecurityGroupEgress
	ec2:AuthorizeSecurityGroupIngress
	ec2:BundleInstance
	ec2:CancelBundleTask
	ec2:CancelCapacityReservation
	ec2:CancelCapacityReservationFleets
	ec2:CancelConversionTask
	ec2:CancelDeclarativePoliciesReport
	ec2:CancelExportTask
	ec2:CancelImageLaunchPermission
	ec2:CancelImportTask
	ec2:CancelReservedInstancesListing
	ec2:CancelSpotFleetRequests
	ec2:CancelSpotInstanceRequests
	ec2:ConfirmProductInstance
	ec2:CopyFpgaImage
	ec2:CopyImage
	ec2:CopySnapshot
	ec2:CopyVolumes

Prefixo do serviço	Ações do
	ec2:CreateCapacityManagerDataExport
	ec2:CreateCapacityReservation
	ec2:CreateCapacityReservationBySplitting
	ec2:CreateCapacityReservationCancellationQuote
	ec2:CreateCapacityReservationFleet
	ec2:CreateCarrierGateway
	ec2:CreateClientVpnEndpoint
	ec2:CreateClientVpnRoute
	ec2:CreateCoipCidr
	ec2:CreateCoipPool
	ec2:CreateCustomerGateway
	ec2:CreateDefaultSubnet
	ec2:CreateDefaultVpc
	ec2:CreateDelegateMacVolumeOwnershipTask
	ec2:CreateDhcpOptions
	ec2:CreateEgressOnlyInternetGateway
	ec2:CreateFleet
	ec2:CreateFlowLogs
	ec2:CreateFpgaImage
	ec2:CreateImage
	ec2:CreateImageUsageReport

Prefixo do serviço	Ações do
	ec2:CreateInstanceConnectEndpoint
	ec2:CreateInstanceEventWindow
	ec2:CreateInstanceExportTask
	ec2:CreateInternetGateway
	ec2:CreateInterruptibleCapacityReservationAllocation
	ec2:CreateIpam
	ec2:CreateIpamExternalResourceVerificationToken
	ec2:CreateIpamPolicy
	ec2:CreateIpamPool
	ec2:CreateIpamPrefixListResolver
	ec2:CreateIpamPrefixListResolverTarget
	ec2:CreateIpamResourceDiscovery
	ec2:CreateIpamScope
	ec2:CreateKeyPair
	ec2:CreateLaunchTemplateVersion
	ec2:CreateLocalGatewayRoute
	ec2:CreateLocalGatewayRouteTable
	ec2:CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:CreateLocalGatewayRouteTableVpcAssociation
	ec2:CreateLocalGatewayVirtualInterface

Prefixo do serviço	Ações do
	ec2:CreateLocalGatewayVirtualInterfaceGroup
	ec2:CreateMacSystemIntegrityProtectionModificationTask
	ec2:CreateManagedPrefixList
	ec2:CreateNatGateway
	ec2:CreateNetworkAcl
	ec2:CreateNetworkAclEntry
	ec2:CreateNetworkInsightsAccessScope
	ec2:CreateNetworkInsightsPath
	ec2:CreateNetworkInterface
	ec2:CreateNetworkInterfacePermission
	ec2:CreatePlacementGroup
	ec2:CreatePublicIpv4Pool
	ec2:CreateReplaceRootVolumeTask
	ec2:CreateReservedInstancesListing
	ec2:CreateRestoreImageTask
	ec2:CreateRoute
	ec2:CreateRouteServer
	ec2:CreateRouteServerEndpoint
	ec2:CreateRouteServerPeer
	ec2:CreateRouteTable
	ec2:CreateSecondaryNetwork

Prefixo do serviço	Ações do
	ec2:CreateSecondarySubnet
	ec2:CreateSecurityGroup
	ec2:CreateSnapshots
	ec2:CreateSpotDatafeedSubscription
	ec2:CreateStoreImageTask
	ec2:CreateSubnet
	ec2:CreateSubnetCidrReservation
	ec2:CreateTrafficMirrorFilter
	ec2:CreateTrafficMirrorFilterRule
	ec2:CreateTrafficMirrorSession
	ec2:CreateTrafficMirrorTarget
	ec2:CreateTransitGateway
	ec2:CreateTransitGatewayConnect
	ec2:CreateTransitGatewayConnectPeer
	ec2:CreateTransitGatewayMeteringPolicy
	ec2:CreateTransitGatewayMeteringPolicyEntry
	ec2:CreateTransitGatewayMulticastDomain
	ec2:CreateTransitGatewayPeeringAttachment
	ec2:CreateTransitGatewayPolicyTable
	ec2:CreateTransitGatewayPrefixListReference
	ec2:CreateTransitGatewayRoute

Prefixo do serviço	Ações do
	ec2:CreateTransitGatewayRouteTable
	ec2:CreateTransitGatewayRouteTableAnnouncement
	ec2:CreateTransitGatewayVpcAttachment
	ec2:CreateVerifiedAccessEndpoint
	ec2:CreateVerifiedAccessGroup
	ec2:CreateVerifiedAccessInstance
	ec2:CreateVerifiedAccessTrustProvider
	ec2:CreateVolume
	ec2:CreateVpc
	ec2:CreateVpcBlockPublicAccessExclusion
	ec2:CreateVpcEncryptionControl
	ec2:CreateVpcEndpoint
	ec2:CreateVpcEndpointConnectionNotification
	ec2:CreateVpcEndpointServiceConfiguration
	ec2:CreateVpcPeeringConnection
	ec2:CreateVpnConcentrator
	ec2:CreateVpnConnection
	ec2:CreateVpnConnectionRoute
	ec2:CreateVpnGateway
	ec2:DeleteCapacityManagerDataExport
	ec2:DeleteCarrierGateway

Prefixo do serviço	Ações do
	ec2:DeleteClientVpnEndpoint
	ec2:DeleteClientVpnRoute
	ec2:DeleteCoipCidr
	ec2:DeleteCoipPool
	ec2:DeleteCustomerGateway
	ec2:DeleteDhcpOptions
	ec2:DeleteEgressOnlyInternetGateway
	ec2:DeleteFleets
	ec2:DeleteFlowLogs
	ec2:DeleteFpgaImage
	ec2:DeleteImageUsageReport
	ec2:DeleteInstanceConnectEndpoint
	ec2:DeleteInstanceEventWindow
	ec2:DeleteInternetGateway
	ec2:DeleteIpam
	ec2:DeleteIpamExternalResourceVerificationToken
	ec2:DeleteIpamPolicy
	ec2:DeleteIpamPool
	ec2:DeleteIpamPrefixListResolver
	ec2:DeleteIpamPrefixListResolverTarget
	ec2:DeleteIpamResourceDiscovery

Prefixo do serviço	Ações do
	ec2:DeleteIpamScope
	ec2:DeleteKeyPair
	ec2:DeleteLaunchTemplate
	ec2:DeleteLaunchTemplateVersions
	ec2:DeleteLocalGatewayRoute
	ec2:DeleteLocalGatewayRouteTable
	ec2:DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:DeleteLocalGatewayRouteTableVpcAssociation
	ec2:DeleteLocalGatewayVirtualInterface
	ec2:DeleteLocalGatewayVirtualInterfaceGroup
	ec2:DeleteManagedPrefixList
	ec2:DeleteNatGateway
	ec2:DeleteNetworkAcl
	ec2:DeleteNetworkAclEntry
	ec2:DeleteNetworkInsightsAccessScope
	ec2:DeleteNetworkInsightsAccessScopeAnalysis
	ec2:DeleteNetworkInsightsAnalysis
	ec2:DeleteNetworkInsightsPath
	ec2:DeleteNetworkInterface
	ec2:DeleteNetworkInterfacePermission

Prefixo do serviço	Ações do
	ec2:DeletePlacementGroup
	ec2:DeletePublicIpv4Pool
	ec2:DeleteQueuedReservedInstances
	ec2:DeleteRoute
	ec2:DeleteRouteServer
	ec2:DeleteRouteServerEndpoint
	ec2:DeleteRouteServerPeer
	ec2:DeleteRouteTable
	ec2:DeleteSecondaryNetwork
	ec2:DeleteSecondarySubnet
	ec2:DeleteSecurityGroup
	ec2:DeleteSpotDatafeedSubscription
	ec2:DeleteSubnet
	ec2:DeleteSubnetCidrReservation
	ec2:DeleteTrafficMirrorFilter
	ec2:DeleteTrafficMirrorFilterRule
	ec2:DeleteTrafficMirrorSession
	ec2:DeleteTrafficMirrorTarget
	ec2:DeleteTransitGateway
	ec2:DeleteTransitGatewayClientVpnAttachment
	ec2:DeleteTransitGatewayConnect

Prefixo do serviço	Ações do
	ec2:DeleteTransitGatewayConnectPeer
	ec2:DeleteTransitGatewayMeteringPolicy
	ec2:DeleteTransitGatewayMeteringPolicyEntry
	ec2:DeleteTransitGatewayMulticastDomain
	ec2:DeleteTransitGatewayPeeringAttachment
	ec2:DeleteTransitGatewayPolicyTable
	ec2:DeleteTransitGatewayPrefixListReference
	ec2:DeleteTransitGatewayRoute
	ec2:DeleteTransitGatewayRouteTable
	ec2:DeleteTransitGatewayRouteTableAnnouncement
	ec2:DeleteTransitGatewayVpcAttachment
	ec2:DeleteVerifiedAccessEndpoint
	ec2:DeleteVerifiedAccessGroup
	ec2:DeleteVerifiedAccessInstance
	ec2:DeleteVerifiedAccessTrustProvider
	ec2:DeleteVolume
	ec2:DeleteVpc
	ec2:DeleteVpcBlockPublicAccessExclusion
	ec2:DeleteVpcEncryptionControl
	ec2:DeleteVpcEndpointConnectionNotifications
	ec2:DeleteVpcEndpointServiceConfigurations

Prefixo do serviço	Ações do
	ec2:DeleteVpcEndpoints
	ec2:DeleteVpcPeeringConnection
	ec2:DeleteVpnConcentrator
	ec2:DeleteVpnConnection
	ec2:DeleteVpnConnectionRoute
	ec2:DeleteVpnGateway
	ec2:DeprovisionByoipCidr
	ec2:DeprovisionIpamByoasn
	ec2:DeprovisionIpamPoolCidr
	ec2:DeprovisionPublicIpv4PoolCidr
	ec2:DeregisterImage
	ec2:DeregisterInstanceEventNotificationAttributes
	ec2:DeregisterTransitGatewayMulticastGroupMembers
	ec2:DeregisterTransitGatewayMulticastGroupSources
	ec2:DescribeAccountAttributes
	ec2:DescribeAddressTransfers
	ec2:DescribeAddresses
	ec2:DescribeAddressesAttribute
	ec2:DescribeAggregateIdFormat
	ec2:DescribeAvailabilityZones
	ec2:DescribeAwsNetworkPerformanceMetricSubscriptions

Prefixo do serviço	Ações do
	ec2:DescribeBundleTasks
	ec2:DescribeByoipCidrs
	ec2:DescribeCapacityBlockExtensionHistory
	ec2:DescribeCapacityBlockExtensionOfferings
	ec2:DescribeCapacityBlockStatus
	ec2:DescribeCapacityBlocks
	ec2:DescribeCapacityManagerDataExports
	ec2:DescribeCapacityReservationBillingRequests
	ec2:DescribeCapacityReservationCancellationQuotes
	ec2:DescribeCapacityReservationFleets
	ec2:DescribeCapacityReservationTopology
	ec2:DescribeCapacityReservations
	ec2:DescribeCarrierGateways
	ec2:DescribeClassicLinkInstances
	ec2:DescribeClientVpnAuthorizationRules
	ec2:DescribeClientVpnConnections
	ec2:DescribeClientVpnEndpoints
	ec2:DescribeClientVpnRoutes
	ec2:DescribeClientVpnTargetNetworks
	ec2:DescribeCoipPools
	ec2:DescribeConversionTasks

Prefixo do serviço	Ações do
	ec2:DescribeCustomerGateways
	ec2:DescribeDeclarativePoliciesReports
	ec2:DescribeDhcpOptions
	ec2:DescribeEgressOnlyInternetGateways
	ec2:DescribeElasticGpus
	ec2:DescribeExportImageTasks
	ec2:DescribeExportTasks
	ec2:DescribeFastLaunchImages
	ec2:DescribeFastSnapshotRestores
	ec2:DescribeFleetHistory
	ec2:DescribeFleetInstances
	ec2:DescribeFleets
	ec2:DescribeFlowLogs
	ec2:DescribeFpgaImageAttribute
	ec2:DescribeFpgaImages
	ec2:DescribeHostReservationOfferings
	ec2:DescribeHostReservations
	ec2:DescribeHosts
	ec2:DescribeIamInstanceProfileAssociations
	ec2:DescribeIdFormat
	ec2:DescribeIdentityIdFormat

Prefixo do serviço	Ações do
	ec2:DescribeImageAttribute
	ec2:DescribeImageReferences
	ec2:DescribeImageUsageReportEntries
	ec2:DescribeImageUsageReports
	ec2:DescribeImportImageTasks
	ec2:DescribeImportSnapshotTasks
	ec2:DescribeInstanceConnectEndpoints
	ec2:DescribeInstanceCreditSpecifications
	ec2:DescribeInstanceEventNotificationAttributes
	ec2:DescribeInstanceEventWindows
	ec2:DescribeInstanceImageMetadata
	ec2:DescribeInstanceSqlHaHistoryStates
	ec2:DescribeInstanceSqlHaStates
	ec2:DescribeInstanceTopology
	ec2:DescribeInstanceTypes
	ec2:DescribeInternetGateways
	ec2:DescribeIpamByoasn
	ec2:DescribeIpamExternalResourceVerificationTokens
	ec2:DescribeIpamPolicies
	ec2:DescribeIpamPoolAllocations
	ec2:DescribeIpamPools

Prefixo do serviço	Ações do
	ec2:DescribeIpamPrefixListResolverTargets
	ec2:DescribeIpamPrefixListResolvers
	ec2:DescribeIpamResourceDiscoveries
	ec2:DescribeIpamResourceDiscoveryAssociations
	ec2:DescribeIpamScopes
	ec2:DescribeIpams
	ec2:DescribeIpv6Pools
	ec2:DescribeKeyPairs
	ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	ec2:DescribeLocalGatewayRouteTableVpcAssociations
	ec2:DescribeLocalGatewayRouteTables
	ec2:DescribeLocalGatewayVirtualInterfaceGroups
	ec2:DescribeLocalGatewayVirtualInterfaces
	ec2:DescribeLocalGateways
	ec2:DescribeLockedSnapshots
	ec2:DescribeMacHosts
	ec2:DescribeMacModificationTasks
	ec2:DescribeManagedPrefixLists
	ec2:DescribeMovingAddresses
	ec2:DescribeNatGateways

Prefixo do serviço	Ações do
	ec2:DescribeNetworkAcls
	ec2:DescribeNetworkInsightsAccessScopeAnalyses
	ec2:DescribeNetworkInsightsAccessScopes
	ec2:DescribeNetworkInsightsAnalyses
	ec2:DescribeNetworkInsightsPaths
	ec2:DescribeNetworkInterfaceAttribute
	ec2:DescribeNetworkInterfacePermissions
	ec2:DescribeNetworkInterfaces
	ec2:DescribeOutpostLags
	ec2:DescribePlacementGroups
	ec2:DescribePrefixLists
	ec2:DescribePrincipalIdFormat
	ec2:DescribePublicIpv4Pools
	ec2:DescribeRegions
	ec2:DescribeReplaceRootVolumeTasks
	ec2:DescribeReservedInstances
	ec2:DescribeReservedInstancesListings
	ec2:DescribeReservedInstancesModifications
	ec2:DescribeReservedInstancesOfferings
	ec2:DescribeRouteServerEndpoints
	ec2:DescribeRouteServerPeers

Prefixo do serviço	Ações do
	ec2:DescribeRouteServers
	ec2:DescribeRouteTables
	ec2:DescribeScheduledInstanceAvailability
	ec2:DescribeScheduledInstances
	ec2:DescribeSecondaryInterfaces
	ec2:DescribeSecondaryNetworks
	ec2:DescribeSecondarySubnets
	ec2:DescribeSecurityGroupReferences
	ec2:DescribeSecurityGroupRules
	ec2:DescribeSecurityGroupVpcAssociations
	ec2:DescribeSecurityGroups
	ec2:DescribeServiceLinkVirtualInterfaces
	ec2:DescribeSnapshotAttribute
	ec2:DescribeSnapshotTierStatus
	ec2:DescribeSpotDatafeedSubscription
	ec2:DescribeSpotFleetInstances
	ec2:DescribeSpotFleetRequestHistory
	ec2:DescribeSpotFleetRequests
	ec2:DescribeSpotInstanceRequests
	ec2:DescribeSpotPriceHistory
	ec2:DescribeStaleSecurityGroups

Prefixo do serviço	Ações do
	ec2:DescribeStoreImageTasks
	ec2:DescribeTrafficMirrorFilterRules
	ec2:DescribeTrafficMirrorFilters
	ec2:DescribeTrafficMirrorSessions
	ec2:DescribeTrafficMirrorTargets
	ec2:DescribeTransitGatewayAttachments
	ec2:DescribeTransitGatewayConnectPeers
	ec2:DescribeTransitGatewayConnects
	ec2:DescribeTransitGatewayMeteringPolicies
	ec2:DescribeTransitGatewayMulticastDomains
	ec2:DescribeTransitGatewayPeeringAttachments
	ec2:DescribeTransitGatewayPolicyTables
	ec2:DescribeTransitGatewayRouteTableAnnouncements
	ec2:DescribeTransitGatewayRouteTables
	ec2:DescribeTransitGatewayVpcAttachments
	ec2:DescribeTransitGateways
	ec2:DescribeTrunkInterfaceAssociations
	ec2:DescribeVerifiedAccessEndpoints
	ec2:DescribeVerifiedAccessGroups
	ec2:DescribeVerifiedAccessInstanceLoggingConfigurations
	ec2:DescribeVerifiedAccessInstances

Prefixo do serviço	Ações do
	ec2:DescribeVerifiedAccessTrustProviders
	ec2:DescribeVolumeAttribute
	ec2:DescribeVolumeStatus
	ec2:DescribeVolumes
	ec2:DescribeVolumesModifications
	ec2:DescribeVpcAttribute
	ec2:DescribeVpcBlockPublicAccessExclusions
	ec2:DescribeVpcBlockPublicAccessOptions
	ec2:DescribeVpcClassicLink
	ec2:DescribeVpcClassicLinkDnsSupport
	ec2:DescribeVpcEncryptionControls
	ec2:DescribeVpcEndpointAssociations
	ec2:DescribeVpcEndpointConnectionNotifications
	ec2:DescribeVpcEndpointConnections
	ec2:DescribeVpcEndpointServiceConfigurations
	ec2:DescribeVpcEndpointServicePermissions
	ec2:DescribeVpcEndpointServices
	ec2:DescribeVpcEndpoints
	ec2:DescribeVpcPeeringConnections
	ec2:DescribeVpcs
	ec2:DescribeVpnConcentrators

Prefixo do serviço	Ações do
	ec2:DescribeVpnConnections
	ec2:DescribeVpnGateways
	ec2:DetachClassicLinkVpc
	ec2:DetachImageWatermark
	ec2:DetachInternetGateway
	ec2:DetachNetworkInterface
	ec2:DetachVerifiedAccessTrustProvider
	ec2:DetachVolume
	ec2:DetachVpnGateway
	ec2:DisableAddressTransfer
	ec2:DisableAllowedImagesSettings
	ec2:DisableAwsNetworkPerformanceMetricSubscription
	ec2:DisableCapacityManager
	ec2:DisableEbsEncryptionByDefault
	ec2:DisableFastLaunch
	ec2:DisableFastSnapshotRestores
	ec2:DisableImage
	ec2:DisableImageBlockPublicAccess
	ec2:DisableImageDeprecation
	ec2:DisableImageDeregistrationProtection
	ec2:DisableInstanceSqlHaStandbyDetections

Prefixo do serviço	Ações do
	ec2:DisableIpamOrganizationAdminAccount
	ec2:DisableIpamPolicy
	ec2:DisableRouteServerPropagation
	ec2:DisableSerialConsoleAccess
	ec2:DisableSnapshotBlockPublicAccess
	ec2:DisableTransitGatewayRouteTablePropagation
	ec2:DisableVgwRoutePropagation
	ec2:DisableVpcClassicLink
	ec2:DisableVpcClassicLinkDnsSupport
	ec2:DisassociateAddress
	ec2:DisassociateCapacityReservationBillingOwner
	ec2:DisassociateClientVpnTargetNetwork
	ec2:DisassociateEnclaveCertificateIamRole
	ec2:DisassociateIamInstanceProfile
	ec2:DisassociateInstanceEventWindow
	ec2:DisassociateIpamByoasn
	ec2:DisassociateIpamResourceDiscovery
	ec2:DisassociateNatGatewayAddress
	ec2:DisassociateRouteServer
	ec2:DisassociateRouteTable
	ec2:DisassociateSecurityGroupVpc

Prefixo do serviço	Ações do
	ec2:DisassociateSubnetCidrBlock
	ec2:DisassociateTransitGatewayMulticastDomain
	ec2:DisassociateTransitGatewayPolicyTable
	ec2:DisassociateTransitGatewayRouteTable
	ec2:DisassociateTrunkInterface
	ec2:DisassociateVpcCidrBlock
	ec2:EnableAddressTransfer
	ec2:EnableAllowedImagesSettings
	ec2:EnableAwsNetworkPerformanceMetricSubscription
	ec2:EnableCapacityManager
	ec2:EnableEbsEncryptionByDefault
	ec2:EnableFastLaunch
	ec2:EnableFastSnapshotRestores
	ec2:EnableImage
	ec2:EnableImageBlockPublicAccess
	ec2:EnableImageDeprecation
	ec2:EnableImageDeregistrationProtection
	ec2:EnableInstanceSqlHaStandbyDetections
	ec2:EnableIpamOrganizationAdminAccount
	ec2:EnableIpamPolicy
	ec2:EnableReachabilityAnalyzerOrganizationSharing

Prefixo do serviço	Ações do
	ec2:EnableRouteServerPropagation
	ec2:EnableSerialConsoleAccess
	ec2:EnableSnapshotBlockPublicAccess
	ec2:EnableTransitGatewayRouteTablePropagation
	ec2:EnableVgwRoutePropagation
	ec2:EnableVolumeIO
	ec2:EnableVpcClassicLink
	ec2:EnableVpcClassicLinkDnsSupport
	ec2:ExportClientVpnClientCertificateRevocationList
	ec2:ExportClientVpnClientConfiguration
	ec2:ExportImage
	ec2:ExportTransitGatewayRoutes
	ec2:ExportVerifiedAccessInstanceClientConfiguration
	ec2:GetActiveVpnTunnelStatus
	ec2:GetAllowedImagesSettings
	ec2:GetAssociatedEnclaveCertificateIamRoles
	ec2:GetAssociatedIpv6PoolCidrs
	ec2:GetAwsNetworkPerformanceData
	ec2:GetCapacityManagerAttributes
	ec2:GetCapacityManagerMetricData
	ec2:GetCapacityManagerMetricDimensions

Prefixo do serviço	Ações do
	ec2:GetCapacityReservationUsage
	ec2:GetCoipPoolUsage
	ec2:GetConsoleOutput
	ec2:GetConsoleScreenshot
	ec2:GetDeclarativePoliciesReportSummary
	ec2:GetDefaultCreditSpecification
	ec2:GetEbsDefaultKmsKeyId
	ec2:GetEbsEncryptionByDefault
	ec2:GetEnabledIpamPolicy
	ec2:GetFlowLogsIntegrationTemplate
	ec2:GetGroupsForCapacityReservation
	ec2:GetHostReservationPurchasePreview
	ec2:GetImageAncestry
	ec2:GetImageBlockPublicAccessState
	ec2:GetInstanceMetadataDefaults
	ec2:GetInstanceTpmEkPub
	ec2:GetInstanceTypesFromInstanceRequirements
	ec2:GetInstanceUefiData
	ec2:GetIpamAddressHistory
	ec2:GetIpamDiscoveredAccounts
	EC2: Obtenha endereços públicos descobertos pelo IPAM

Prefixo do serviço	Ações do
	ec2:GetIpamDiscoveredResourceCidrs
	ec2:GetIpamPolicyAllocationRules
	ec2:GetIpamPolicyOrganizationTargets
	ec2:GetIpamPoolAllocations
	ec2:GetIpamPoolCidrs
	ec2:GetIpamPrefixListResolverRules
	ec2:GetIpamPrefixListResolverVersionEntries
	ec2:GetIpamPrefixListResolverVersions
	ec2:GetIpamResourceCidrs
	ec2:GetLaunchTemplateData
	ec2:GetManagedPrefixListAssociations
	ec2:GetManagedPrefixListEntries
	ec2:GetManagedResourceVisibility
	ec2:GetNetworkInsightsAccessScopeAnalysisFindings
	ec2:GetNetworkInsightsAccessScopeContent
	ec2:GetPasswordData
	ec2:GetReservedInstancesExchangeQuote
	ec2:GetRouteServerAssociations
	ec2:GetRouteServerPropagations
	ec2:GetRouteServerRoutingDatabase
	ec2:GetSecurityGroupsForVpc

Prefixo do serviço	Ações do
	ec2:GetSerialConsoleAccessStatus
	ec2:GetSnapshotBlockPublicAccessState
	ec2:GetSpotPlacementScores
	ec2:GetSubnetCidrReservations
	ec2:GetTransitGatewayAttachmentPropagations
	ec2:GetTransitGatewayMeteringPolicyEntries
	ec2:GetTransitGatewayMulticastDomainAssociations
	ec2:GetTransitGatewayPolicyTableAssociations
	ec2:GetTransitGatewayPolicyTableEntries
	ec2:GetTransitGatewayPrefixListReferences
	ec2:GetTransitGatewayRouteTableAssociations
	ec2:GetTransitGatewayRouteTablePropagations
	ec2:GetVerifiedAccessEndpointPolicy
	ec2:GetVerifiedAccessEndpointTargets
	ec2:GetVerifiedAccessGroupPolicy
	ec2:GetVpcResourcesBlockingEncryptionEnforcement
	ec2:GetVpnConnectionDeviceSampleConfiguration
	ec2:GetVpnConnectionDeviceTypes
	ec2:GetVpnTunnelReplacementStatus
	ec2:ImportClientVpnClientCertificateRevocationList
	ec2:ImportImage

Prefixo do serviço	Ações do
	ec2:ImportInstance
	ec2:ImportKeyPair
	ec2:ImportSnapshot
	ec2:ImportVolume
	ec2:ListImagesInRecycleBin
	ec2:ListSnapshotsInRecycleBin
	ec2:ListVolumesInRecycleBin
	ec2:LockSnapshot
	ec2:ModifyAddressAttribute
	ec2:ModifyAvailabilityZoneGroup
	ec2:ModifyCapacityReservation
	ec2:ModifyCapacityReservationFleet
	ec2:ModifyClientVpnEndpoint
	ec2:ModifyDefaultCreditSpecification
	ec2:ModifyEbsDefaultKmsKeyId
	ec2:ModifyFleet
	ec2:ModifyFpgaImageAttribute
	ec2:ModifyHosts
	ec2:ModifyIdFormat
	ec2:ModifyIdentityIdFormat
	ec2:ModifyImageAttribute

Prefixo do serviço	Ações do
	ec2:ModifyInstanceAttribute
	ec2:ModifyInstanceCapacityReservationAttributes
	ec2:ModifyInstanceConnectEndpoint
	ec2:ModifyInstanceCpuOptions
	ec2:ModifyInstanceCreditSpecification
	ec2:ModifyInstanceEventStartTime
	ec2:ModifyInstanceEventWindow
	ec2:ModifyInstanceMaintenanceOptions
	ec2:ModifyInstanceMetadataDefaults
	ec2:ModifyInstanceMetadataOptions
	ec2:ModifyInstanceNetworkPerformanceOptions
	ec2:ModifyInstancePlacement
	ec2:ModifyIpam
	ec2:ModifyIpamPolicyAllocationRules
	ec2:ModifyIpamPool
	ec2:ModifyIpamPoolAllocation
	ec2:ModifyIpamPrefixListResolver
	ec2:ModifyIpamPrefixListResolverTarget
	ec2:ModifyIpamResourceCidr
	ec2:ModifyIpamResourceDiscovery
	ec2:ModifyIpamScope

Prefixo do serviço	Ações do
	ec2:ModifyLaunchTemplate
	ec2:ModifyLocalGatewayRoute
	ec2:ModifyManagedPrefixList
	ec2:ModifyManagedResourceVisibility
	ec2:ModifyNetworkInterfaceAttribute
	ec2:ModifyPrivateDnsNameOptions
	ec2:ModifyPublicIpDnsNameOptions
	ec2:ModifyReservedInstances
	ec2:ModifyRouteServer
	ec2:ModifySecurityGroupRules
	ec2:ModifySnapshotAttribute
	ec2:ModifySnapshotTier
	ec2:ModifySpotFleetRequest
	ec2:ModifySubnetAttribute
	ec2:ModifyTrafficMirrorFilterNetworkServices
	ec2:ModifyTrafficMirrorFilterRule
	ec2:ModifyTrafficMirrorSession
	ec2:ModifyTransitGateway
	ec2:ModifyTransitGatewayMeteringPolicy
	ec2:ModifyTransitGatewayPrefixListReference
	ec2:ModifyTransitGatewayVpcAttachment

Prefixo do serviço	Ações do
	ec2:ModifyVerifiedAccessEndpoint
	ec2:ModifyVerifiedAccessEndpointPolicy
	ec2:ModifyVerifiedAccessGroup
	ec2:ModifyVerifiedAccessGroupPolicy
	ec2:ModifyVerifiedAccessInstance
	ec2:ModifyVerifiedAccessInstanceLoggingConfiguration
	ec2:ModifyVerifiedAccessTrustProvider
	ec2:ModifyVolume
	ec2:ModifyVolumeAttribute
	ec2:ModifyVpcAttribute
	ec2:ModifyVpcBlockPublicAccessExclusion
	ec2:ModifyVpcBlockPublicAccessOptions
	ec2:ModifyVpcEncryptionControl
	ec2:ModifyVpcEndpoint
	ec2:ModifyVpcEndpointConnectionNotification
	ec2:ModifyVpcEndpointServiceConfiguration
	ec2:ModifyVpcEndpointServicePayerResponsibility
	ec2:ModifyVpcEndpointServicePermissions
	ec2:ModifyVpcPeeringConnectionOptions
	ec2:ModifyVpcTenancy
	ec2:ModifyVpnConnection

Prefixo do serviço	Ações do
	ec2:ModifyVpnConnectionOptions
	ec2:ModifyVpnTunnelCertificate
	ec2:ModifyVpnTunnelOptions
	ec2:MonitorInstances
	ec2:MoveAddressToVpc
	ec2:MoveByoipCidrToIpam
	ec2:MoveCapacityReservationInstances
	ec2:ProvisionByoipCidr
	ec2:ProvisionIpamByoasn
	ec2:ProvisionIpamPoolCidr
	ec2:ProvisionPublicIpv4PoolCidr
	ec2:PurchaseCapacityBlockExtension
	ec2:PurchaseHostReservation
	ec2:PurchaseReservedInstancesOffering
	ec2:PurchaseScheduledInstances
	ec2:RebootInstances
	ec2:RegisterImage
	ec2:RegisterInstanceEventNotificationAttributes
	ec2:RegisterTransitGatewayMulticastGroupMembers
	ec2:RegisterTransitGatewayMulticastGroupSources
	ec2:RejectCapacityReservationBillingOwnership

Prefixo do serviço	Ações do
	ec2:RejectTransitGatewayClientVpnAttachment
	ec2:RejectTransitGatewayMulticastDomainAssociations
	ec2:RejectTransitGatewayPeeringAttachment
	ec2:RejectTransitGatewayVpcAttachment
	ec2:RejectVpcEndpointConnections
	ec2:RejectVpcPeeringConnection
	ec2:ReleaseAddress
	ec2:ReleaseHosts
	ec2:ReleaseIpamPoolAllocation
	ec2:ReplaceIamInstanceProfileAssociation
	ec2:ReplaceImageCriteriaInAllowedImagesSettings
	ec2:ReplaceNetworkAclAssociation
	ec2:ReplaceNetworkAclEntry
	ec2:ReplaceRoute
	ec2:ReplaceRouteTableAssociation
	ec2:ReplaceTransitGatewayRoute
	ec2:ReplaceVpnTunnel
	ec2:ReportInstanceStatus
	ec2:RequestSpotFleet
	ec2:RequestSpotInstances
	ec2:ResetAddressAttribute

Prefixo do serviço	Ações do
	ec2:ResetEbsDefaultKmsKeyId
	ec2:ResetFpgaImageAttribute
	ec2:ResetImageAttribute
	ec2:ResetInstanceAttribute
	ec2:ResetNetworkInterfaceAttribute
	ec2:ResetSnapshotAttribute
	ec2:RestoreAddressToClassic
	ec2:RestoreImageFromRecycleBin
	ec2:RestoreManagedPrefixListVersion
	ec2:RestoreSnapshotFromRecycleBin
	ec2:RestoreSnapshotTier
	ec2:RestoreVolumeFromRecycleBin
	ec2:RevokeClientVpnIngress
	ec2:RevokeSecurityGroupEgress
	ec2:RevokeSecurityGroupIngress
	ec2:RunInstances
	ec2:RunScheduledInstances
	ec2:SearchLocalGatewayRoutes
	ec2:SearchTransitGatewayMulticastGroups
	ec2:SearchTransitGatewayRoutes
	ec2:SendDiagnosticInterrupt

Prefixo do serviço	Ações do
	ec2:StartDeclarativePoliciesReport
	ec2:StartInstances
	ec2:StartNetworkInsightsAccessScopeAnalysis
	ec2:StartNetworkInsightsAnalysis
	ec2:StartVpcEndpointServicePrivateDnsVerification
	ec2:TerminateClientVpnConnections
	ec2:UnassignIpv6Addresses
	ec2:UnassignPrivateIpAddresses
	ec2:UnassignPrivateNatGatewayAddress
	ec2:UnlockSnapshot
	ec2:UnmonitorInstances
	ec2:UpdateCapacityManagerOrganizationsAccess
	ec2:UpdateInterruptibleCapacityReservationAllocation
	ec2:UpdateSecurityGroupRuleDescriptionsEgress
	ec2:UpdateSecurityGroupRuleDescriptionsIngress
	ec2:WithdrawByoipCidr

Prefixo do serviço	Ações do
ecr	ecr:BatchCheckLayerAvailability ecr:BatchDeleteImage ecr:BatchGetImage ecr:BatchGetRepositoryScanningConfiguration ecr:CompleteLayerUpload ecr:CreatePullThroughCacheRule ecr:CreateRepositoryCreationTemplate ecr:DeleteLifecyclePolicy ecr:DeletePullThroughCacheRule ecr:DeleteRegistryPolicy ecr:DeleteRepository ecr:DeleteRepositoryCreationTemplate ecr:DeleteRepositoryPolicy ecr:DeleteSigningConfiguration ecr:DescribeImageReplicationStatus ecr:DescribeImageScanFindings ecr:DescribeImages ecr:DescribePullThroughCacheRules ecr:DescribeRegistry ecr:DescribeRepositories ecr:DescribeRepositoryCreationTemplates

Prefixo do serviço	Ações do
	ecr:GetAccountSetting ecr:GetAuthorizationToken ecr:GetDownloadUrlForLayer ecr:GetLifecyclePolicy ecr:GetLifecyclePolicyPreview ecr:GetRegistryPolicy ecr:GetRegistryScanningConfiguration ecr:GetRepositoryPolicy ecr:GetSigningConfiguration ecr:InitiateLayerUpload ecr:ListImages ecr:ListPullTimeUpdateExclusions ecr:PutAccountSetting ecr:PutImage ecr:PutImageScanningConfiguration ecr:PutRegistryPolicy ecr:PutRegistryScanningConfiguration ecr:PutReplicationConfiguration ecr:StartImageScan ecr:StartLifecyclePolicyPreview ecr:UpdatePullThroughCacheRule

Prefixo do serviço	Ações do
	ecr:UpdateRepositoryCreationTemplate ecr:UploadLayerPart ecr:ValidatePullThroughCacheRule

Prefixo do serviço	Ações do
ecr-public	ecr-public:BatchCheckLayerAvailability
	ecr-public:BatchDeleteImage
	ecr-public:CompleteLayerUpload
	ecr-public:CreateRepository
	ecr-public:DeleteRepository
	ecr-public:DeleteRepositoryPolicy
	ecr-public:DescribeImages
	ecr-public:DescribeRegistries
	ecr-public:DescribeRepositories
	ecr-public:GetAuthorizationToken
	ecr-public:GetRegistryCatalogData
	ecr-public:GetRepositoryCatalogData
	ecr-public:GetRepositoryPolicy
	ecr-public:InitiateLayerUpload
	ecr-public:PutImage
	ecr-public:PutRegistryCatalogData
	ecr-public:PutRepositoryCatalogData
	ecr-public:SetRepositoryPolicy
	ecr-public:UploadLayerPart

Prefixo do serviço	Ações do
ecs	ecs:ContinueServiceDeployment ecs:CreateCapacityProvider ecs:CreateCluster ecs:CreateDaemon ecs:CreateService ecs:CreateTaskSet ecs:DeleteAccountSetting ecs:DeleteAttributes ecs:DeleteCapacityProvider ecs:DeleteCluster ecs:DeleteDaemon ecs:DeleteDaemonTaskDefinition ecs:DeleteExpressGatewayService ecs:DeleteService ecs:DeleteTaskDefinitions ecs:DeleteTaskSet ecs:DeregisterContainerInstance ecs:DeregisterTaskDefinition ecs:DescribeCapacityProviders ecs:DescribeClusters ecs:DescribeContainerInstances

Prefixo do serviço	Ações do
	ecs:DescribeDaemon ecs:DescribeDaemonDeployments ecs:DescribeDaemonRevisions ecs:DescribeDaemonTaskDefinition ecs:DescribeExpressGatewayService ecs:DescribeServiceDeployments ecs:DescribeServiceRevisions ecs:DescribeServices ecs:DescribeTaskDefinition ecs:DescribeTaskSets ecs:DescribeTasks ecs:DiscoverPollEndpoint ecs:ExecuteCommand ecs:GetTaskProtection ecs:ListAccountSettings ecs:ListAttributes ecs:ListClusters ecs:ListContainerInstances ecs:ListDaemonDeployments ecs:ListDaemonTaskDefinitions ecs:ListDaemons

Prefixo do serviço	Ações do
	ecs:ListServiceDeployments ecs:ListServices ecs:ListServicesByNamespace ecs:ListTaskDefinitionFamilies ecs:ListTaskDefinitions ecs:ListTasks ecs:PutAccountSetting ecs:PutAccountSettingDefault ecs:PutAttributes ecs:PutClusterCapacityProviders ecs:RegisterContainerInstance ecs:RegisterDaemonTaskDefinition ecs:RunTask ecs:StartTask ecs:StopServiceDeployment ecs:StopTask ecs:SubmitAttachmentStateChanges ecs:SubmitContainerStateChange ecs:SubmitTaskStateChange ecs:UpdateCapacityProvider ecs:UpdateCluster

Prefixo do serviço	Ações do
	ecs:UpdateClusterSettings ecs:UpdateContainerAgent ecs:UpdateContainerInstancesState ecs:UpdateDaemon ecs:UpdateExpressGatewayService ecs:UpdateService ecs:UpdateServicePrimaryTaskSet ecs:UpdateTaskProtection ecs:UpdateTaskSet

Prefixo do serviço	Ações do
eks	eks:AssociateAccessPolicy
	eks:AssociateEncryptionConfig
	eks:AssociateIdentityProviderConfig
	eks:CancelUpdate
	eks:CreateAccessEntry
	eks:CreateAddon
	eks:CreateCluster
	eks:CreateEksAnywhereSubscription
	eks:CreateFargateProfile
	eks:CreateNodegroup
	eks:DeleteAccessEntry
	eks:DeleteAddon
	eks:DeleteCapability
	eks:DeleteCluster
	eks:DeleteEksAnywhereSubscription
	eks:DeleteFargateProfile
	eks:DeleteNodegroup
	eks:DeletePodIdentityAssociation
	eks:DeregisterCluster
	eks:DescribeAccessEntry
	eks:DescribeAddon

Prefixo do serviço	Ações do
	eks:DescribeAddonConfiguration
	eks:DescribeAddonVersions
	eks:DescribeCapability
	eks:DescribeCluster
	eks:DescribeClusterVersions
	eks:DescribeEksAnywhereSubscription
	eks:DescribeFargateProfile
	eks:DescribeIdentityProviderConfig
	eks:DescribeInsight
	eks:DescribeInsightsRefresh
	eks:DescribeNodegroup
	eks:DescribePodIdentityAssociation
	eks:DescribeUpdate
	eks:DisassociateAccessPolicy
	eks:DisassociateIdentityProviderConfig
	eks:ListAccessEntries
	eks:ListAccessPolicies
	eks:ListAddons
	eks:ListAssociatedAccessPolicies
	eks:ListCapabilities
	eks:ListClusters

Prefixo do serviço	Ações do
	eks:ListEksAnywhereSubscriptions
	eks:ListFargateProfiles
	eks:ListIdentityProviderConfigs
	eks:ListInsights
	eks:ListNodegroups
	eks:ListPodIdentityAssociations
	eks:ListUpdates
	eks:RegisterCluster
	eks:StartInsightsRefresh
	eks:UpdateAccessEntry
	eks:UpdateAddon
	eks:UpdateCapability
	eks:UpdateClusterConfig
	eks:UpdateClusterVersion
	eks:UpdateEksAnywhereSubscription
	eks:UpdateNodegroupConfig
	eks:UpdateNodegroupVersion
	eks:UpdatePodIdentityAssociation

Prefixo do serviço	Ações do
elasticache	elasticache:AuthorizeCacheSecurityGroupIngress elasticache:BatchApplyUpdateAction elasticache:BatchStopUpdateAction elasticache:CompleteMigration elasticache:CopyServerlessCacheSnapshot elasticache:CopySnapshot elasticache:CreateCacheCluster elasticache:CreateCacheParameterGroup elasticache:CreateCacheSecurityGroup elasticache:CreateCacheSubnetGroup elasticache:CreateGlobalReplicationGroup elasticache:CreateReplicationGroup elasticache:CreateServerlessCache elasticache:CreateServerlessCacheSnapshot elasticache:CreateSnapshot elasticache:CreateUser elasticache:CreateUserGroup elasticache:DecreaseNodeGroupsInGlobalReplicationGroup elasticache:DecreaseReplicaCount elasticache:DeleteCacheCluster elasticache:DeleteCacheParameterGroup

Prefixo do serviço	Ações do
	elasticache:DeleteCacheSecurityGroup elasticache:DeleteCacheSubnetGroup elasticache:DeleteGlobalReplicationGroup elasticache:DeleteReplicationGroup elasticache:DeleteServerlessCache elasticache:DeleteServerlessCacheSnapshot elasticache:DeleteSnapshot elasticache:DeleteUser elasticache:DeleteUserGroup elasticache:DescribeCacheClusters elasticache:DescribeCacheEngineVersions elasticache:DescribeCacheParameterGroups elasticache:DescribeCacheParameters elasticache:DescribeCacheSecurityGroups elasticache:DescribeCacheSubnetGroups elasticache:DescribeEngineDefaultParameters elasticache:DescribeEvents elasticache:DescribeGlobalReplicationGroups elasticache:DescribeReplicationGroups elasticache:DescribeReservedCacheNodes elasticache:DescribeReservedCacheNodesOfferings

Prefixo do serviço	Ações do
	elasticache:DescribeServerlessCacheSnapshots elasticache:DescribeServerlessCaches elasticache:DescribeServiceUpdates elasticache:DescribeSnapshots elasticache:DescribeUpdateActions elasticache:DescribeUserGroups elasticache:DescribeUsers elasticache:DisassociateGlobalReplicationGroup elasticache:ExportServerlessCacheSnapshot elasticache:FailoverGlobalReplicationGroup elasticache:IncreaseNodeGroupsInGlobalReplicationGroup elasticache:IncreaseReplicaCount elasticache:ListAllowedNodeTypeModifications elasticache:ModifyCacheCluster elasticache:ModifyCacheParameterGroup elasticache:ModifyCacheSubnetGroup elasticache:ModifyGlobalReplicationGroup elasticache:ModifyReplicationGroup elasticache:ModifyReplicationGroupShardConfiguration elasticache:ModifyServerlessCache elasticache:ModifyUser

Prefixo do serviço	Ações do
	elasticache:ModifyUserGroup elasticache:PurchaseReservedCacheNodesOffering elasticache:RebalanceSlotsInGlobalReplicationGroup elasticache:RebootCacheCluster elasticache:ResetCacheParameterGroup elasticache:RevokeCacheSecurityGroupIngress elasticache:StartMigration elasticache:TestFailover elasticache:TestMigration

Prefixo do serviço	Ações do
elasticbeanstalk	elasticbeanstalk:AbortEnvironmentUpdate
	elasticbeanstalk:ApplyEnvironmentManagedAction
	elasticbeanstalk:AssociateEnvironmentOperationsRole
	elasticbeanstalk:CheckDNSAvailability
	elasticbeanstalk:ComposeEnvironments
	elasticbeanstalk:CreateApplication
	elasticbeanstalk:CreateApplicationVersion
	elasticbeanstalk:CreateConfigurationTemplate
	elasticbeanstalk:CreateEnvironment
	elasticbeanstalk:CreatePlatformVersion
	elasticbeanstalk:CreateStorageLocation
	elasticbeanstalk:DeleteApplication
	elasticbeanstalk:DeleteApplicationVersion
	elasticbeanstalk:DeleteConfigurationTemplate
	elasticbeanstalk:DeleteEnvironmentConfiguration
	elasticbeanstalk:DeletePlatformVersion
	elasticbeanstalk:DescribeAccountAttributes
	elasticbeanstalk:DescribeApplicationVersions
	elasticbeanstalk:DescribeApplications
	elasticbeanstalk:DescribeConfigurationOptions
	elasticbeanstalk:DescribeConfigurationSettings

Prefixo do serviço	Ações do
	elasticbeanstalk:DescribeEnvironmentHealth
	elasticbeanstalk:DescribeEnvironmentManagedActionHistory
	elasticbeanstalk:DescribeEnvironmentManagedActions
	elasticbeanstalk:DescribeEnvironmentResources
	elasticbeanstalk:DescribeEnvironments
	elasticbeanstalk:DescribeEvents
	elasticbeanstalk:DescribeInstancesHealth
	elasticbeanstalk:DescribePlatformVersion
	elasticbeanstalk:DisassociateEnvironmentOperationsRole
	elasticbeanstalk:ListAvailableSolutionStacks
	elasticbeanstalk:ListPlatformBranches
	elasticbeanstalk:ListPlatformVersions
	elasticbeanstalk:RebuildEnvironment
	elasticbeanstalk:RequestEnvironmentInfo
	elasticbeanstalk:RestartAppServer
	elasticbeanstalk:RetrieveEnvironmentInfo
	elasticbeanstalk:SwapEnvironmentCNAMEs
	elasticbeanstalk:TerminateEnvironment
	elasticbeanstalk:UpdateApplication
	elasticbeanstalk:UpdateApplicationResourceLifecycle
	elasticbeanstalk:UpdateApplicationVersion

Prefixo do serviço	Ações do
	elasticbeanstalk:UpdateConfigurationTemplate elasticbeanstalk:UpdateEnvironment elasticbeanstalk:ValidateConfigurationSettings

Prefixo do serviço	Ações do
elasticfilesystem	elasticfilesystem:CreateAccessPoint elasticfilesystem:CreateFileSystem elasticfilesystem:CreateMountTarget elasticfilesystem:CreateReplicationConfiguration elasticfilesystem:DeleteAccessPoint elasticfilesystem:DeleteFileSystem elasticfilesystem:DeleteFileSystemPolicy elasticfilesystem:DeleteMountTarget elasticfilesystem:DeleteReplicationConfiguration elasticfilesystem:DescribeAccessPoints elasticfilesystem:DescribeAccountPreferences elasticfilesystem:DescribeBackupPolicy elasticfilesystem:DescribeFileSystemPolicy elasticfilesystem:DescribeFileSystems elasticfilesystem:DescribeLifecycleConfiguration elasticfilesystem:DescribeMountTargetSecurityGroups elasticfilesystem:DescribeMountTargets elasticfilesystem:DescribeReplicationConfigurations elasticfilesystem:ModifyMountTargetSecurityGroups elasticfilesystem:PutAccountPreferences elasticfilesystem:PutBackupPolicy

Prefixo do serviço	Ações do
	elasticfilesystem:PutFileSystemPolicy elasticfilesystem:PutLifecycleConfiguration elasticfilesystem:UpdateFileSystem elasticfilesystem:UpdateFileSystemProtection

Prefixo do serviço	Ações do
elasticloadbalancing	elasticloadbalancing:AddListenerCertificates
	elasticloadbalancing:AddTrustStoreRevocations
	elasticloadbalancing:ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing:AttachLoadBalancerToSubnets
	elasticloadbalancing:ConfigureHealthCheck
	elasticloadbalancing:CreateAppCookieStickinessPolicy
	elasticloadbalancing:CreateLBCookieStickinessPolicy
	elasticloadbalancing:CreateListener
	elasticloadbalancing:CreateLoadBalancer
	elasticloadbalancing:CreateLoadBalancerListeners
	elasticloadbalancing:CreateLoadBalancerPolicy
	elasticloadbalancing:CreateRule
	elasticloadbalancing:CreateTargetGroup
	elasticloadbalancing:CreateTrustStore
	elasticloadbalancing:DeleteListener
	elasticloadbalancing:DeleteLoadBalancer
	elasticloadbalancing:DeleteLoadBalancerListeners
	elasticloadbalancing:DeleteLoadBalancerPolicy
	elasticloadbalancing:DeleteRule
	elasticloadbalancing:DeleteSharedTrustStoreAssociation
	elasticloadbalancing:DeleteTargetGroup

Prefixo do serviço	Ações do
	elasticloadbalancing:DeleteTrustStore
	elasticloadbalancing:DeregisterInstancesFromLoadBalancer
	elasticloadbalancing:DeregisterTargets
	elasticloadbalancing:DescribeAccountLimits
	elasticloadbalancing:DescribeCapacityReservation
	elasticloadbalancing:DescribeInstanceHealth
	elasticloadbalancing:DescribeListenerAttributes
	elasticloadbalancing:DescribeListenerCertificates
	elasticloadbalancing:DescribeListeners
	elasticloadbalancing:DescribeLoadBalancerAttributes
	elasticloadbalancing:DescribeLoadBalancerPolicies
	elasticloadbalancing:DescribeLoadBalancerPolicyTypes
	elasticloadbalancing:DescribeLoadBalancers
	elasticloadbalancing:DescribeRules
	elasticloadbalancing:DescribeSSLPolicies
	elasticloadbalancing:DescribeTargetGroupAttributes
	elasticloadbalancing:DescribeTargetGroups
	elasticloadbalancing:DescribeTargetHealth
	elasticloadbalancing:DescribeTrustStoreAssociations
	elasticloadbalancing:DescribeTrustStoreRevocations
	elasticloadbalancing:DescribeTrustStores

Prefixo do serviço	Ações do
	elasticloadbalancing:DetachLoadBalancerFromSubnets
	elasticloadbalancing:DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:GetResourcePolicy
	elasticloadbalancing:GetTrustStoreCaCertificatesBundle
	elasticloadbalancing:GetTrustStoreRevocationContent
	elasticloadbalancing:ModifyCapacityReservation
	elasticloadbalancing:ModifyIpPools
	elasticloadbalancing:ModifyListener
	elasticloadbalancing:ModifyLoadBalancerAttributes
	elasticloadbalancing:ModifyRule
	elasticloadbalancing:ModifyTargetGroup
	elasticloadbalancing:ModifyTargetGroupAttributes
	elasticloadbalancing:ModifyTrustStore
	elasticloadbalancing:RegisterInstancesWithLoadBalancer
	elasticloadbalancing:RegisterTargets
	elasticloadbalancing:RemoveListenerCertificates
	elasticloadbalancing:RemoveTrustStoreRevocations
	elasticloadbalancing:SetIpAddressType
	elasticloadbalancing:SetLoadBalancerListenerSSLCertificate
	elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer

Prefixo do serviço	Ações do
	elasticloadbalancing:SetLoadBalancerPoliciesOfListener elasticloadbalancing:SetRulePriorities elasticloadbalancing:SetSecurityGroups elasticloadbalancing:SetSubnets

Prefixo do serviço	Ações do
emr-containers	emr-containers:CancelJobRun emr-containers:CreateJobTemplate emr-containers:CreateManagedEndpoint emr-containers:CreateSecurityConfiguration emr-containers:CreateVirtualCluster emr-containers:DeleteJobTemplate emr-containers:DeleteManagedEndpoint emr-containers:DeleteSecurityConfiguration emr-containers:DeleteVirtualCluster emr-containers:DescribeJobRun emr-containers:DescribeJobTemplate emr-containers:DescribeManagedEndpoint emr-containers:DescribeSecurityConfiguration emr-containers:DescribeVirtualCluster emr-containers:GetManagedEndpointSessionCredentials emr-containers:ListJobRuns emr-containers:ListJobTemplates emr-containers:ListManagedEndpoints emr-containers:ListSecurityConfigurations emr-containers:ListVirtualClusters emr-containers:StartJobRun

Prefixo do serviço	Ações do
emr-serverless	emr-serverless:CancelJobRun
	emr-serverless:CreateApplication
	emr-serverless:DeleteApplication
	emr-serverless:GetApplication
	emr-serverless:GetDashboardForJobRun
	emr-serverless:GetJobRun
	emr-serverless:GetResourceDashboard
	emr-serverless:GetSession
	emr-serverless:GetSessionEndpoint
	emr-serverless:ListApplications
	emr-serverless:ListJobRunAttempts
	emr-serverless:ListJobRuns
	emr-serverless:ListSessions
	emr-serverless:StartApplication
	emr-serverless:StartJobRun
	emr-serverless:StopApplication
	emr-serverless:TerminateSession
	emr-serverless:UpdateApplication

Prefixo do serviço	Ações do
es	es:AcceptInboundConnection es:AcceptInboundCrossClusterSearchConnection es:AssociatePackage es:AuthorizeVpcEndpointAccess es:CancelElasticsearchServiceSoftwareUpdate es:CancelServiceSoftwareUpdate es:CreateDomain es:CreateElasticsearchDomain es:CreateIndex es:CreateOutboundConnection es:CreateOutboundCrossClusterSearchConnection es:CreatePackage es:CreateVpcEndpoint es:DeleteDomain es:DeleteElasticsearchDomain es:DeleteElasticsearchServiceRole es:DeleteInboundConnection es:DeleteInboundCrossClusterSearchConnection es:DeleteIndex es:DeleteOutboundConnection es:DeleteOutboundCrossClusterSearchConnection

Prefixo do serviço	Ações do
	es:DeletePackage es:DeleteVpcEndpoint es:DescribeDomain es:DescribeDomainAutoTunes es:DescribeDomainChangeProgress es:DescribeDomainConfig es:DescribeDomainHealth es:DescribeDomainNodes es:DescribeDomains es:DescribeDryRunProgress es:DescribeElasticsearchDomain es:DescribeElasticsearchDomainConfig es:DescribeElasticsearchDomains es:DescribeElasticsearchInstanceTypeLimits es:DescribeInboundConnections es:DescribeInboundCrossClusterSearchConnections es:DescribeInstanceTypeLimits es:DescribeOutboundConnections es:DescribeOutboundCrossClusterSearchConnections es:DescribePackages es:DescribeReservedElasticsearchInstanceOfferings

Prefixo do serviço	Ações do
	es:DescribeReservedElasticsearchInstances es:DescribeReservedInstanceOfferings es:DescribeReservedInstances es:DescribeVpcEndpoints es:DissociatePackage es:DissociatePackages es:GetCompatibleElasticsearchVersions es:GetCompatibleVersions es:GetDataSource es:GetDomainMaintenanceStatus es:GetPackageVersionHistory es:GetUpgradeHistory es:GetUpgradeStatus es:ListDataSources es:ListDomainNames es:ListDomainsForPackage es:ListElasticsearchInstanceTypes es:ListElasticsearchVersions es:ListInsights es:ListInstanceTypeDetails es:ListPackagesForDomain

Prefixo do serviço	Ações do
	es:ListScheduledActions es:ListVersions es:ListVpcEndpointAccess es:ListVpcEndpoints es:ListVpcEndpointsForDomain es:PurchaseReservedElasticsearchInstanceOffering es:PurchaseReservedInstanceOffering es:RejectInboundConnection es:RejectInboundCrossClusterSearchConnection es:RevokeVpcEndpointAccess es:RollbackServiceSoftwareUpdate es:StartDomainMaintenance es:StartElasticsearchServiceSoftwareUpdate es:StartServiceSoftwareUpdate es:UpdateDataSource es:UpdateDomainConfig es:UpdateElasticsearchDomainConfig es:UpdateIndex es:UpdatePackage es:UpdatePackageScope es:UpdateScheduledAction

Prefixo do serviço	Ações do
	es:UpdateVpcEndpoint es:UpgradeDomain es:UpgradeElasticsearchDomain

Prefixo do serviço	Ações do
eventos	events:ActivateEventSource
	events:CancelReplay
	events:CreateApiDestination
	events:CreateArchive
	events:CreateConnection
	events:CreateEndpoint
	events:CreateEventBus
	events:CreatePartnerEventSource
	events:DeactivateEventSource
	events:DeauthorizeConnection
	events:DeleteApiDestination
	events:DeleteArchive
	events:DeleteConnection
	events:DeleteEndpoint
	events:DeleteEventBus
	events:DeletePartnerEventSource
	events:DeleteRule
	events:DescribeApiDestination
	events:DescribeArchive
	events:DescribeConnection
	events:DescribeEndpoint

Prefixo do serviço	Ações do
	events:DescribeEventBus
	events:DescribeEventSource
	events:DescribePartnerEventSource
	events:DescribeReplay
	events:DescribeRule
	events:DisableRule
	events:EnableRule
	events:ListApiDestinations
	events:ListArchives
	events:ListConnections
	events:ListEndpoints
	events:ListEventBuses
	events:ListEventSources
	events:ListPartnerEventSourceAccounts
	events:ListPartnerEventSources
	events:ListReplays
	events:ListRuleNamesByTarget
	events:ListRules
	events:ListTargetsByRule
	events:PutPermission
	events:PutRule

Prefixo do serviço	Ações do
	events:PutTargets
	events:RemovePermission
	events:RemoveTargets
	events:StartReplay
	events:TestEventPattern
	events:UpdateApiDestination
	events:UpdateArchive
	events:UpdateConnection
	events:UpdateEndpoint
	events:UpdateEventBus

Prefixo do serviço	Ações do
finspace	finspace:CreateEnvironment finspace:CreateKxChangeset finspace:CreateKxCluster finspace:CreateKxDatabase finspace:CreateKxDataview finspace:CreateKxEnvironment finspace:CreateKxScalingGroup finspace:CreateKxUser finspace:CreateKxVolume finspace:CreateUser finspace:DeleteEnvironment finspace:DeleteKxCluster finspace:DeleteKxClusterNode finspace:DeleteKxDatabase finspace:DeleteKxDataview finspace:DeleteKxEnvironment finspace:DeleteKxScalingGroup finspace:DeleteKxUser finspace:DeleteKxVolume finspace:GetEnvironment finspace:GetKxChangeset

Prefixo do serviço	Ações do
	finspace:GetKxCluster finspace:GetKxConnectionString finspace:GetKxDatabase finspace:GetKxDataview finspace:GetKxEnvironment finspace:GetKxScalingGroup finspace:GetKxUser finspace:GetKxVolume finspace:GetLoadSampleDataSetGroupIntoEnvironmentStatus finspace:GetUser finspace:ListEnvironments finspace:ListKxChangesets finspace:ListKxClusterNodes finspace:ListKxClusters finspace:ListKxDatabases finspace:ListKxDataviews finspace:ListKxEnvironments finspace:ListKxScalingGroups finspace:ListKxUsers finspace:ListKxVolumes finspace:ListUsers

Prefixo do serviço	Ações do
	finspace:LoadSampleDataSetGroupIntoEnvironment finspace:ResetUserPassword finspace:UpdateEnvironment finspace:UpdateKxClusterCodeConfiguration finspace:UpdateKxClusterDatabases finspace:UpdateKxDatabase finspace:UpdateKxDataview finspace:UpdateKxEnvironment finspace:UpdateKxEnvironmentNetwork finspace:UpdateKxUser finspace:UpdateKxVolume finspace:UpdateUser
firehose	firehose:CreateDeliveryStream firehose:DeleteDeliveryStream firehose:DescribeDeliveryStream firehose:ListDeliveryStreams firehose:StartDeliveryStreamEncryption firehose:StopDeliveryStreamEncryption firehose:UpdateDestination

Prefixo do serviço	Ações do
fis	fis:CreateExperimentTemplate fis:CreateTargetAccountConfiguration fis:DeleteExperimentTemplate fis:DeleteTargetAccountConfiguration fis:GetAction fis:GetExperiment fis:GetExperimentTargetAccountConfiguration fis:GetExperimentTemplate fis:GetSafetyLever fis:GetTargetAccountConfiguration fis:GetTargetResourceType fis:ListActions fis:ListExperimentResolvedTargets fis:ListExperimentTargetAccountConfigurations fis:ListExperimentTemplates fis:ListExperiments fis:ListTargetAccountConfigurations fis:ListTargetResourceTypes fis:StartExperiment fis:StopExperiment fis:UpdateExperimentTemplate

Prefixo do serviço	Ações do
	fis:UpdateSafetyLeverState fis:UpdateTargetAccountConfiguration

Prefixo do serviço	Ações do
fms	fms:AssociateAdminAccount
	fms:AssociateThirdPartyFirewall
	fms:BatchAssociateResource
	fms:BatchDisassociateResource
	fms:DeleteAppsList
	fms:DeleteNotificationChannel
	fms:DeletePolicy
	fms:DeleteProtocolsList
	fms:DeleteResourceSet
	fms:DisassociateAdminAccount
	fms:DisassociateThirdPartyFirewall
	fms:GetAdminAccount
	fms:GetAdminScope
	fms:GetAppsList
	fms:GetComplianceDetail
	fms:GetNotificationChannel
	fms:GetPolicy
	fms:GetProtectionStatus
	fms:GetProtocolsList
	fms:GetResourceSet
	fms:GetThirdPartyFirewallAssociationStatus

Prefixo do serviço	Ações do
	fms:GetViolationDetails
	fms:ListAdminAccountsForOrganization
	fms:ListAdminsManagingAccount
	fms:ListAppsLists
	fms:ListComplianceStatus
	fms:ListDiscoveredResources
	fms:ListMemberAccounts
	fms:ListPolicies
	fms:ListProtocolsLists
	fms:ListResourceSetResources
	fms:ListResourceSets
	fms:ListThirdPartyFirewallFirewallPolicies
	fms:PutAdminAccount
	fms:PutAppsList
	fms:PutNotificationChannel
	fms:PutPolicy
	fms:PutProtocolsList
	fms:PutResourceSet

Prefixo do serviço	Ações do
frauddetector	frauddetector:BatchCreateVariable frauddetector:BatchGetVariable frauddetector:CancelBatchImportJob frauddetector:CancelBatchPredictionJob frauddetector:CreateBatchImportJob frauddetector:CreateBatchPredictionJob frauddetector:CreateDetectorVersion frauddetector:CreateList frauddetector:CreateModel frauddetector:CreateModelVersion frauddetector:CreateRule frauddetector:CreateVariable frauddetector:DeleteBatchImportJob frauddetector:DeleteBatchPredictionJob frauddetector:DeleteDetector frauddetector:DeleteDetectorVersion frauddetector:DeleteEntityType frauddetector:DeleteEvent frauddetector:DeleteEventType frauddetector:DeleteEventsByEventType frauddetector:DeleteExternalModel

Prefixo do serviço	Ações do
	frauddetector:DeleteLabel frauddetector:DeleteList frauddetector:DeleteModel frauddetector:DeleteModelVersion frauddetector:DeleteOutcome frauddetector:DeleteRule frauddetector:DeleteVariable frauddetector:DescribeDetector frauddetector:DescribeModelVersions frauddetector:GetBatchImportJobs frauddetector:GetBatchPredictionJobs frauddetector:GetDeleteEventsByEventTypeStatus frauddetector:GetDetectorVersion frauddetector:GetDetectors frauddetector:GetEntityTypes frauddetector:GetEvent frauddetector:GetEventPrediction frauddetector:GetEventPredictionMetadata frauddetector:GetEventTypes frauddetector:GetExternalModels frauddetector:GetKMSEncryptionKey

Prefixo do serviço	Ações do
	frauddetector:GetLabels frauddetector:GetListElements frauddetector:GetListsMetadata frauddetector:GetModelVersion frauddetector:GetModels frauddetector:GetOutcomes frauddetector:GetRules frauddetector:GetVariables frauddetector:ListEventPredictions frauddetector:PutDetector frauddetector:PutEntityType frauddetector:PutEventType frauddetector:PutExternalModel frauddetector:PutKMSEncryptionKey frauddetector:PutLabel frauddetector:PutOutcome frauddetector:SendEvent frauddetector:UpdateDetectorVersion frauddetector:UpdateDetectorVersionMetadata frauddetector:UpdateDetectorVersionStatus frauddetector:UpdateEventLabel

Prefixo do serviço	Ações do
	frauddetector:UpdateList frauddetector:UpdateModel frauddetector:UpdateModelVersion frauddetector:UpdateModelVersionStatus frauddetector:UpdateRuleMetadata frauddetector:UpdateRuleVersion frauddetector:UpdateVariable

Prefixo do serviço	Ações do
fsx	fsx:AssociateFileSystemAliases
	fsx:CancelDataRepositoryTask
	fsx:CopyBackup
	fsx:CreateDataRepositoryTask
	fsx:CreateFileCache
	fsx:CreateFileSystem
	fsx:CreateFileSystemFromBackup
	fsx:CreateSnapshot
	fsx:CreateStorageVirtualMachine
	fsx:CreateVolume
	fsx:CreateVolumeFromBackup
	fsx:DeleteBackup
	fsx:DeleteFileCache
	fsx:DeleteFileSystem
	fsx:DeleteSnapshot
	fsx:DeleteStorageVirtualMachine
	fsx:DeleteVolume
	fsx:DescribeBackups
	fsx:DescribeDataRepositoryAssociations
	fsx:DescribeDataRepositoryTasks
	fsx:DescribeFileCaches

Prefixo do serviço	Ações do
	fsx:DescribeFileSystemAliases
	fsx:DescribeFileSystems
	fsx:DescribeS3AccessPointAttachments
	fsx:DescribeSharedVpcConfiguration
	fsx:DescribeSnapshots
	fsx:DescribeStorageVirtualMachines
	fsx:DescribeVolumes
	fsx:DetachAndDeleteS3AccessPoint
	fsx:DisassociateFileSystemAliases
	fsx:ReleaseFileSystemNfsV3Locks
	fsx:RestoreVolumeFromSnapshot
	fsx:StartMisconfiguredStateRecovery
	fsx:UpdateDataRepositoryAssociation
	fsx:UpdateFileCache
	fsx:UpdateFileSystem
	fsx:UpdateSharedVpcConfiguration
	fsx:UpdateSnapshot
	fsx:UpdateStorageVirtualMachine
	fsx:UpdateVolume

Prefixo do serviço	Ações do
gamelift	gamelift:AcceptMatch gamelift:ClaimGameServer gamelift:CreateAlias gamelift:CreateBuild gamelift:CreateContainerGroupDefinition gamelift:CreateFleet gamelift:CreateFleetLocations gamelift:CreateGameServerGroup gamelift:CreateGameSession gamelift:CreateGameSessionQueue gamelift:CreateLocation gamelift:CreateMatchmakingConfiguration gamelift:CreateMatchmakingRuleSet gamelift:CreatePlayerSession gamelift:CreatePlayerSessions gamelift:CreateScript gamelift:CreateVpcPeeringAuthorization gamelift:CreateVpcPeeringConnection gamelift:DeleteAlias gamelift:DeleteBuild gamelift:DeleteContainerGroupDefinition

Prefixo do serviço	Ações do
	gamelift:DeleteFleet gamelift:DeleteFleetLocations gamelift:DeleteGameServerGroup gamelift:DeleteGameSessionQueue gamelift:DeleteLocation gamelift:DeleteMatchmakingConfiguration gamelift:DeleteMatchmakingRuleSet gamelift:DeleteScalingPolicy gamelift:DeleteScript gamelift:DeleteVpcPeeringAuthorization gamelift:DeleteVpcPeeringConnection gamelift:DeregisterCompute gamelift:DeregisterGameServer gamelift:DescribeAlias gamelift:DescribeBuild gamelift:DescribeCompute gamelift:DescribeContainerFleet gamelift:DescribeContainerGroupDefinition gamelift:DescribeContainerGroupPortMappings gamelift:DescribeEC2InstanceLimits gamelift:DescribeFleetAttributes

Prefixo do serviço	Ações do
	gamelift:DescribeFleetCapacity gamelift:DescribeFleetEvents gamelift:DescribeFleetLocationAttributes gamelift:DescribeFleetLocationCapacity gamelift:DescribeFleetLocationUtilization gamelift:DescribeFleetPortSettings gamelift:DescribeFleetUtilization gamelift:DescribeGameServer gamelift:DescribeGameServerGroup gamelift:DescribeGameServerInstances gamelift:DescribeGameSessionDetails gamelift:DescribeGameSessionPlacement gamelift:DescribeGameSessionQueues gamelift:DescribeGameSessions gamelift:DescribeInstances gamelift:DescribeMatchmaking gamelift:DescribeMatchmakingConfigurations gamelift:DescribeMatchmakingRuleSets gamelift:DescribePlayerSessions gamelift:DescribeRuntimeConfiguration gamelift:DescribeScalingPolicies

Prefixo do serviço	Ações do
	gamelift:DescribeScript gamelift:DescribeVpcPeeringAuthorizations gamelift:DescribeVpcPeeringConnections gamelift:GetComputeAccess gamelift:GetComputeAuthToken gamelift:GetGameSessionLogUrl gamelift:GetInstanceAccess gamelift:GetPlayerConnectionDetails gamelift:ListAliases gamelift:ListBuilds gamelift:ListCompute gamelift:ListContainerFleets gamelift:ListContainerGroupDefinitionVersions gamelift:ListContainerGroupDefinitions gamelift:ListFleetDeployments gamelift:ListFleets gamelift:ListGameServerGroups gamelift:ListGameServers gamelift:ListLocations gamelift:ListScripts gamelift:PutScalingPolicy

Prefixo do serviço	Ações do
	gamelift:RegisterCompute gamelift:RegisterGameServer gamelift:RequestUploadCredentials gamelift:ResolveAlias gamelift:ResumeGameServerGroup gamelift:SearchGameSessions gamelift:StartFleetActions gamelift:StartGameSessionPlacement gamelift:StartMatchBackfill gamelift:StartMatchmaking gamelift:StopFleetActions gamelift:StopGameSessionPlacement gamelift:StopMatchmaking gamelift:SuspendGameServerGroup gamelift:TerminateGameSession gamelift:UpdateAlias gamelift:UpdateBuild gamelift:UpdateContainerGroupDefinition gamelift:UpdateFleetAttributes gamelift:UpdateFleetCapacity gamelift:UpdateFleetPortSettings

Prefixo do serviço	Ações do
	gamelift:UpdateGameServer gamelift:UpdateGameServerGroup gamelift:UpdateGameSession gamelift:UpdateGameSessionQueue gamelift:UpdateMatchmakingConfiguration gamelift:UpdateRuntimeConfiguration gamelift:UpdateScript gamelift:ValidateMatchmakingRuleSet

Prefixo do serviço	Ações do
geo	geo:AssociateTrackerConsumer
	geo:BatchDeleteDevicePositionHistory
	geo:BatchDeleteGeofence
	geo:BatchEvaluateGeofences
	geo:BatchGetDevicePosition
	geo:BatchPutGeofence
	geo:BatchUpdateDevicePosition
	geo:CalculateRoute
	geo:CalculateRouteMatrix
	geo:CancelJob
	geo:CreateGeofenceCollection
	geo:CreateMap
	geo:CreatePlaceIndex
	geo:CreateRouteCalculator
	geo:CreateTracker
	geo:DeleteGeofenceCollection
	geo:DeleteKey
	geo:DeleteMap
	geo:DeletePlaceIndex
	geo:DeleteRouteCalculator
	geo:DeleteTracker

Prefixo do serviço	Ações do
	geo:DescribeGeofenceCollection
	geo:DescribeKey
	geo:DescribeMap
	geo:DescribePlaceIndex
	geo:DescribeRouteCalculator
	geo:DescribeTracker
	geo:DisassociateTrackerConsumer
	geo:ForecastGeofenceEvents
	geo:GetDevicePosition
	geo:GetDevicePositionHistory
	geo:GetGeofence
	geo:GetJob
	geo:GetMapGlyphs
	geo:GetMapSprites
	geo:GetMapStyleDescriptor
	geo:GetMapTile
	geo:GetPlace
	geo:ListDevicePositions
	geo:ListGeofenceCollections
	geo:ListGeofences
	geo:ListJobs

Prefixo do serviço	Ações do
	geo:ListKeys
	geo:ListMaps
	geo:ListPlaceIndexes
	geo:ListRouteCalculators
	geo:ListTrackerConsumers
	geo:ListTrackers
	geo:PutGeofence
	geo:SearchPlaceIndexForPosition
	geo:SearchPlaceIndexForSuggestions
	geo:SearchPlaceIndexForText
	geo:UpdateGeofenceCollection
	geo:UpdateKey
	geo:UpdateMap
	geo:UpdatePlaceIndex
	geo:UpdateRouteCalculator
	geo:UpdateTracker
	geo:VerifyDevicePosition

Prefixo do serviço	Ações do
glacier	glacier:AbortMultipartUpload glacier:AbortVaultLock glacier:CompleteMultipartUpload glacier:CompleteVaultLock glacier:CreateVault glacier:DeleteArchive glacier:DeleteVault glacier:DeleteVaultAccessPolicy glacier:DeleteVaultNotifications glacier:DescribeJob glacier:DescribeVault glacier:GetDataRetrievalPolicy glacier:GetJobOutput glacier:GetVaultAccessPolicy glacier:GetVaultLock glacier:GetVaultNotifications glacier:InitiateJob glacier:InitiateMultipartUpload glacier:InitiateVaultLock glacier:ListJobs glacier:ListMultipartUploads

Prefixo do serviço	Ações do
	glacier:ListParts glacier:ListProvisionedCapacity glacier:ListVaults glacier:PurchaseProvisionedCapacity glacier:SetDataRetrievalPolicy glacier:SetVaultAccessPolicy glacier:SetVaultNotifications glacier:UploadArchive glacier:UploadMultipartPart

Prefixo do serviço	Ações do
grafana	grafana:AssociateLicense
	grafana:CreateWorkspace
	grafana:CreateWorkspaceApiKey
	grafana:CreateWorkspaceServiceAccount
	grafana:CreateWorkspaceServiceAccountToken
	grafana:DeleteWorkspace
	grafana:DeleteWorkspaceApiKey
	grafana:DeleteWorkspaceServiceAccount
	grafana:DeleteWorkspaceServiceAccountToken
	grafana:DescribeWorkspace
	grafana:DescribeWorkspaceAuthentication
	grafana:DescribeWorkspaceConfiguration
	grafana:DisassociateLicense
	grafana:ListPermissions
	grafana:ListVersions
	grafana:ListWorkspaceServiceAccountTokens
	grafana:ListWorkspaceServiceAccounts
	grafana:ListWorkspaces
	grafana:UpdatePermissions
	grafana:UpdateWorkspace
	grafana:UpdateWorkspaceAuthentication

Prefixo do serviço	Ações do
	grafana:UpdateWorkspaceConfiguration

Prefixo do serviço	Ações do
greengrass	greengrass:AssociateRoleToGroup greengrass:AssociateServiceRoleToAccount greengrass:BatchAssociateClientDeviceWithCoreDevice greengrass:BatchDisassociateClientDeviceFromCoreDevice greengrass:CancelDeployment greengrass:CreateComponentVersion greengrass:CreateConnectorDefinition greengrass:CreateConnectorDefinitionVersion greengrass:CreateCoreDefinition greengrass:CreateCoreDefinitionVersion greengrass:CreateDeployment greengrass:CreateDeviceDefinition greengrass:CreateDeviceDefinitionVersion greengrass:CreateFunctionDefinition greengrass:CreateFunctionDefinitionVersion greengrass:CreateGroup greengrass:CreateGroupCertificateAuthority greengrass:CreateGroupVersion greengrass:CreateLoggerDefinition greengrass:CreateLoggerDefinitionVersion greengrass:CreateResourceDefinition

Prefixo do serviço	Ações do
	greengrass:CreateResourceDefinitionVersion greengrass:CreateSoftwareUpdateJob greengrass:CreateSubscriptionDefinition greengrass:CreateSubscriptionDefinitionVersion greengrass:DeleteComponent greengrass:DeleteConnectorDefinition greengrass:DeleteCoreDefinition greengrass:DeleteCoreDevice greengrass:DeleteDeployment greengrass:DeleteDeviceDefinition greengrass:DeleteFunctionDefinition greengrass:DeleteGroup greengrass:DeleteLoggerDefinition greengrass:DeleteResourceDefinition greengrass:DeleteSubscriptionDefinition greengrass:DescribeComponent greengrass:DisassociateRoleFromGroup greengrass:DisassociateServiceRoleFromAccount greengrass:GetAssociatedRole greengrass:GetBulkDeploymentStatus greengrass:GetComponent

Prefixo do serviço	Ações do
	greengrass:GetComponentVersionArtifact greengrass:GetConnectivityInfo greengrass:GetConnectorDefinition greengrass:GetConnectorDefinitionVersion greengrass:GetCoreDefinition greengrass:GetCoreDefinitionVersion greengrass:GetCoreDevice greengrass:GetDeployment greengrass:GetDeploymentStatus greengrass:GetDeviceDefinition greengrass:GetDeviceDefinitionVersion greengrass:GetFunctionDefinition greengrass:GetFunctionDefinitionVersion greengrass:GetGroup greengrass:GetGroupCertificateAuthority greengrass:GetGroupCertificateConfiguration greengrass:GetGroupVersion greengrass:GetLoggerDefinition greengrass:GetLoggerDefinitionVersion greengrass:GetResourceDefinition greengrass:GetResourceDefinitionVersion

Prefixo do serviço	Ações do
	greengrass:GetServiceRoleForAccount greengrass:GetSubscriptionDefinition greengrass:GetSubscriptionDefinitionVersion greengrass:GetThingRuntimeConfiguration greengrass:ListBulkDeploymentDetailedReports greengrass:ListBulkDeployments greengrass:ListClientDevicesAssociatedWithCoreDevice greengrass:ListComponentVersions greengrass:ListComponents greengrass:ListConnectorDefinitionVersions greengrass:ListConnectorDefinitions greengrass:ListCoreDefinitionVersions greengrass:ListCoreDefinitions greengrass:ListCoreDevices greengrass:ListDeployments greengrass:ListDeviceDefinitionVersions greengrass:ListDeviceDefinitions greengrass:ListEffectiveDeployments greengrass:ListFunctionDefinitionVersions greengrass:ListFunctionDefinitions greengrass:ListGroupCertificateAuthorities

Prefixo do serviço	Ações do
	greengrass:ListGroupVersions greengrass:ListGroups greengrass:ListInstalledComponents greengrass:ListLoggerDefinitionVersions greengrass:ListLoggerDefinitions greengrass:ListResourceDefinitionVersions greengrass:ListResourceDefinitions greengrass:ListSubscriptionDefinitionVersions greengrass:ListSubscriptionDefinitions greengrass:ResetDeployments greengrass:StartBulkDeployment greengrass:StopBulkDeployment greengrass:UpdateConnectivityInfo greengrass:UpdateConnectorDefinition greengrass:UpdateCoreDefinition greengrass:UpdateDeviceDefinition greengrass:UpdateFunctionDefinition greengrass:UpdateGroup greengrass:UpdateGroupCertificateConfiguration greengrass:UpdateLoggerDefinition greengrass:UpdateResourceDefinition

Prefixo do serviço	Ações do
	greengrass:UpdateSubscriptionDefinition greengrass:UpdateThingRuntimeConfiguration

Prefixo do serviço	Ações do
groundstation	groundstation:CancelContact
	groundstation:CreateConfig
	groundstation:CreateDataflowEndpointGroup
	groundstation:CreateDataflowEndpointGroupV2
	groundstation:CreateEphemeris
	groundstation:CreateMissionProfile
	groundstation:DeleteConfig
	groundstation:DeleteDataflowEndpointGroup
	groundstation:DeleteEphemeris
	groundstation:DeleteMissionProfile
	groundstation:DescribeContact
	groundstation:DescribeContactVersion
	groundstation:DescribeEphemeris
	groundstation:GetConfig
	groundstation:GetDataflowEndpointGroup
	groundstation:GetMinuteUsage
	groundstation:GetMissionProfile
	groundstation:GetSatellite
	groundstation:ListAntennas
	groundstation:ListConfigs
	groundstation:ListContacts

Prefixo do serviço	Ações do
	groundstation:ListDataflowEndpointGroups
	groundstation:ListEphemerides
	groundstation:ListGroundStationReservations
	groundstation:ListGroundStations
	groundstation:ListMissionProfiles
	groundstation:ListSatellites
	groundstation:RegisterAgent
	groundstation:ReserveContact
	groundstation:UpdateAgentStatus
	groundstation:UpdateConfig
	groundstation:UpdateEphemeris
	groundstation:UpdateMissionProfile

Prefixo do serviço	Ações do
guardduty	guardduty:AcceptAdministratorInvitation guardduty:AcceptInvitation guardduty:ArchiveFindings guardduty:CreateDetector guardduty:CreateFilter guardduty:CreateIPSet guardduty:CreateInvestigation guardduty:CreateMalwareProtectionPlan guardduty:CreateMembers guardduty:CreatePublishingDestination guardduty:CreateSampleFindings guardduty:CreateThreatEntitySet guardduty:CreateThreatIntelSet guardduty:CreateTrustedEntitySet guardduty:DeclineInvitations guardduty:DeleteDetector guardduty:DeleteFilter guardduty:DeleteIPSet guardduty:DeleteInvitations guardduty:DeleteMalwareProtectionPlan guardduty:DeleteMembers

Prefixo do serviço	Ações do
	guardduty:DeletePublishingDestination guardduty:DeleteThreatEntitySet guardduty:DeleteThreatIntelSet guardduty:DeleteTrustedEntitySet guardduty:DescribeMalwareScans guardduty:DescribeOrganizationConfiguration guardduty:DescribePublishingDestination guardduty:DisableOrganizationAdminAccount guardduty:DisassociateFromAdministratorAccount guardduty:DisassociateFromMasterAccount guardduty:DisassociateMembers guardduty:EnableOrganizationAdminAccount guardduty:GetAdministratorAccount guardduty:GetCoverageStatistics guardduty:GetDetector guardduty:GetFilter guardduty:GetFindings guardduty:GetFindingsStatistics guardduty:GetIPSet guardduty:GetInvestigation guardduty:GetInvitationsCount

Prefixo do serviço	Ações do
	guardduty:GetMalwareProtectionPlan guardduty:GetMalwareScan guardduty:GetMalwareScanSettings guardduty:GetMasterAccount guardduty:GetMemberDetectors guardduty:GetMembers guardduty:GetOrganizationStatistics guardduty:GetRemainingFreeTrialDays guardduty:GetThreatEntitySet guardduty:GetThreatIntelSet guardduty:GetTrustedEntitySet guardduty:GetUsageStatistics guardduty:InviteMembers guardduty:ListCoverage guardduty:ListDetectors guardduty:ListFilters guardduty:ListFindings guardduty:ListIPSets guardduty:ListInvestigations guardduty:ListInvitations guardduty:ListMalwareProtectionPlans

Prefixo do serviço	Ações do
	guardduty:ListMalwareScans guardduty:ListMembers guardduty:ListOrganizationAdminAccounts guardduty:ListPublishingDestinations guardduty:ListThreatEntitySets guardduty:ListThreatIntelSets guardduty:ListTrustedEntitySets guardduty:StartMalwareScan guardduty:StartMonitoringMembers guardduty:StopMonitoringMembers guardduty:UnarchiveFindings guardduty:UpdateDetector guardduty:UpdateFilter guardduty:UpdateFindingsFeedback guardduty:UpdateIPSet guardduty:UpdateMalwareProtectionPlan guardduty:UpdateMalwareScanSettings guardduty:UpdateMemberDetectors guardduty:UpdateOrganizationConfiguration guardduty:UpdatePublishingDestination guardduty:UpdateThreatEntitySet

Prefixo do serviço	Ações do
	guardduty:UpdateThreatIntelSet guardduty:UpdateTrustedEntitySet

Prefixo do serviço	Ações do
healthlake	healthlake:CancelFHIRExportJobWithDelete
	healthlake:CreateDataTransformationProfile
	healthlake:CreateFHIRDatastore
	healthlake:CreateResource
	healthlake:DeleteDataTransformationProfile
	healthlake:DeleteFHIRDatastore
	healthlake:DeleteResource
	healthlake:DescribeDataTransformationJob
	healthlake:DescribeFHIRDatastore
	healthlake:DescribeFHIRExportJob
	healthlake:DescribeFHIRExportJobWithGet
	healthlake:DescribeFHIRImportJob
	healthlake:GetCapabilities
	healthlake:GetDataTransformationProfile
	healthlake:ListDataTransformationJobs
	healthlake:ListDataTransformationProfileVersions
	healthlake:ListDataTransformationProfiles
	healthlake:ListFHIRDatastores
	healthlake:ListFHIRExportJobs
	healthlake:ListFHIRImportJobs
	healthlake:PublishDataTransformationProfile

Prefixo do serviço	Ações do
	healthlake:ReadResource
	healthlake:SearchEverything
	healthlake:SearchWithGet
	healthlake:SearchWithPost
	healthlake:StartFHIRExportJob
	healthlake:StartFHIRExportJobWithPost
	healthlake:StartFHIRImportJob
	healthlake:UpdateDataTransformationProfile
	healthlake:UpdateFHIRDatastore
	healthlake:UpdateProfileWithAgent
	healthlake:UpdateResource

Prefixo do serviço	Ações do
honeycode	honeycode:BatchCreateTableRows honeycode:BatchDeleteTableRows honeycode:BatchUpdateTableRows honeycode:BatchUpsertTableRows honeycode:DescribeTableDataImportJob honeycode:GetScreenData honeycode:InvokeScreenAutomation honeycode:ListTableColumns honeycode:ListTableRows honeycode:ListTables honeycode:QueryTableRows honeycode:StartTableDataImportJob

Prefixo do serviço	Ações do
iam	iam:AddClientIDToOpenIDConnectProvider
	iam:AddRoleToInstanceProfile
	iam:AddUserToGroup
	iam:AttachGroupPolicy
	iam:AttachRolePolicy
	iam:AttachUserPolicy
	iam:ChangePassword
	iam:CreateAccessKey
	iam:CreateAccountAlias
	iam:CreateGroup
	iam:CreateInstanceProfile
	iam:CreateLoginProfile
	iam:CreateOpenIDConnectProvider
	iam:CreatePolicy
	iam:CreatePolicyVersion
	iam:CreateRole
	iam:CreateSAMLProvider
	iam:CreateServiceLinkedRole
	iam:CreateServiceSpecificCredential
	iam:CreateUser
	iam:CreateVirtualMFADevice

Prefixo do serviço	Ações do
	iam:DeactivateMFADevice
	iam:DeleteAccessKey
	iam:DeleteAccountAlias
	iam:DeleteAccountPasswordPolicy
	iam:DeleteCloudFrontPublicKey
	iam:DeleteGroup
	iam:DeleteGroupPolicy
	iam:DeleteInstanceProfile
	iam:DeleteLoginProfile
	iam:DeleteOpenIDConnectProvider
	iam:DeletePolicy
	iam:DeletePolicyVersion
	iam:DeleteRole
	iam:DeleteRolePermissionsBoundary
	iam:DeleteRolePolicy
	iam:DeleteSAMLProvider
	iam:DeleteSSHPublicKey
	iam:DeleteServerCertificate
	iam:DeleteServiceLinkedRole
	iam:DeleteServiceSpecificCredential
	iam:DeleteSigningCertificate

Prefixo do serviço	Ações do
	iam:DeleteUser
	iam:DeleteUserPermissionsBoundary
	iam:DeleteUserPolicy
	iam:DeleteVirtualMFADevice
	iam:DetachGroupPolicy
	iam:DetachRolePolicy
	iam:DetachUserPolicy
	iam:DisableOrganizationsRootCredentialsManagement
	iam:DisableOrganizationsRootSessions
	iam:DisableOutboundWebIdentityFederation
	iam:EnableMFADevice
	iam:EnableOrganizationsRootCredentialsManagement
	iam:EnableOrganizationsRootSessions
	iam:EnableOutboundWebIdentityFederation
	iam:GenerateCredentialReport
	iam:GenerateOrganizationsAccessReport
	iam:GenerateServiceLastAccessedDetails
	iam:GetAccessKeyLastUsed
	iam:GetAccountAuthorizationDetails
	iam:GetAccountEmailAddress
	iam:GetAccountName

Prefixo do serviço	Ações do
	iam:GetAccountPasswordPolicy
	iam:GetAccountSummary
	iam:GetCloudFrontPublicKey
	iam:GetContextKeysForCustomPolicy
	iam:GetContextKeysForPrincipalPolicy
	iam:GetCredentialReport
	iam:GetGroup
	iam:GetGroupPolicy
	iam:GetInstanceProfile
	iam:GetLoginProfile
	iam:GetMFADevice
	iam:GetOpenIDConnectProvider
	iam:GetOrganizationsAccessReport
	iam:GetOutboundWebIdentityFederationInfo
	iam:GetPolicy
	iam:GetPolicyVersion
	iam:GetRole
	iam:GetRolePolicy
	iam:GetSAMLProvider
	iam:GetSSHPublicKey
	iam:GetServerCertificate

Prefixo do serviço	Ações do
	iam:GetServiceLastAccessedDetails
	iam:GetServiceLastAccessedDetailsWithEntities
	iam:GetServiceLinkedRoleDeletionStatus
	iam:GetUser
	iam:GetUserPolicy
	iam:ListAccessKeys
	iam:ListAccountAliases
	iam:ListAttachedGroupPolicies
	iam:ListAttachedRolePolicies
	iam:ListAttachedUserPolicies
	iam:ListCloudFrontPublicKeys
	iam:ListDelegationRequests
	iam:ListEntitiesForPolicy
	iam:ListGroupPolicies
	iam:ListGroups
	iam:ListGroupsForUser
	iam:ListInstanceProfiles
	iam:ListInstanceProfilesForRole
	iam:ListMFADevices
	iam:ListOpenIDConnectProviders
	iam:ListOrganizationsFeatures

Prefixo do serviço	Ações do
	iam:ListPolicies
	iam:ListPoliciesGrantingServiceAccess
	iam:ListPolicyVersions
	iam:ListRolePolicies
	iam:ListRoles
	iam:ListSAMLProviders
	iam:ListSSHPublicKeys
	iam:ListSTSRegionalEndpointsStatus
	iam:ListServerCertificates
	iam:ListServiceSpecificCredentials
	iam:ListSigningCertificates
	iam:ListUserPolicies
	iam:ListUsers
	iam:ListVirtualMFADevices
	iam:PutGroupPolicy
	iam:PutRolePermissionsBoundary
	iam:PutRolePolicy
	iam:PutUserPermissionsBoundary
	iam:PutUserPolicy
	iam:RemoveClientIDFromOpenIDConnectProvider
	iam:RemoveRoleFromInstanceProfile

Prefixo do serviço	Ações do
	iam:RemoveUserFromGroup
	iam:ResetServiceSpecificCredential
	iam:ResyncMFADevice
	iam:SetDefaultPolicyVersion
	iam:SetSTSRegionalEndpointStatus
	iam:SetSecurityTokenServicePreferences
	iam:SimulateCustomPolicy
	iam:SimulatePrincipalPolicy
	iam:UpdateAccessKey
	iam:UpdateAccountEmailAddress
	iam:UpdateAccountName
	iam:UpdateAccountPasswordPolicy
	iam:UpdateAssumeRolePolicy
	iam:UpdateCloudFrontPublicKey
	iam:UpdateGroup
	iam:UpdateLoginProfile
	iam:UpdateOpenIDConnectProviderThumbprint
	iam:UpdateRole
	iam:UpdateRoleDescription
	iam:UpdateSAMLProvider
	iam:UpdateSSHPublicKey

Prefixo do serviço	Ações do
	iam:UpdateServerCertificate
	iam:UpdateServiceSpecificCredential
	iam:UpdateSigningCertificate
	iam:UpdateUser
	iam:UploadCloudFrontPublicKey
	iam:UploadSSHPublicKey
	iam:UploadServerCertificate
	iam:UploadSigningCertificate

Prefixo do serviço	Ações do
identitystore	identitystore:CreateGroup identitystore:CreateGroupMembership identitystore:CreateUser identitystore:DeleteGroup identitystore:DeleteGroupMembership identitystore:DeleteUser identitystore:DescribeGroup identitystore:DescribeGroupMembership identitystore:DescribeUser identitystore:GetGroupId identitystore:GetGroupMembershipId identitystore:GetUserId identitystore:IsMemberInGroups identitystore:ListGroupMemberships identitystore:ListGroupMembershipsForMember identitystore:ListGroups identitystore:ListUsers identitystore:UpdateGroup identitystore:UpdateUser

Prefixo do serviço	Ações do
imagebuilder	imagebuilder:CancelImageCreation
	imagebuilder:CancelLifecycleExecution
	imagebuilder:CreateComponent
	imagebuilder:CreateContainerRecipe
	imagebuilder:CreateDistributionConfiguration
	imagebuilder:CreateImage
	imagebuilder:CreateImagePipeline
	imagebuilder:CreateImageRecipe
	imagebuilder:CreateInfrastructureConfiguration
	imagebuilder:CreateLifecyclePolicy
	imagebuilder:CreateWorkflow
	imagebuilder:DeleteComponent
	imagebuilder:DeleteContainerRecipe
	imagebuilder:DeleteDistributionConfiguration
	imagebuilder:DeleteImage
	imagebuilder:DeleteImagePipeline
	imagebuilder:DeleteImageRecipe
	imagebuilder:DeleteInfrastructureConfiguration
	imagebuilder:DeleteLifecyclePolicy
	imagebuilder:DeleteWorkflow
	imagebuilder:DistributeImage

Prefixo do serviço	Ações do
	imagebuilder:GetComponentPolicy
	imagebuilder:GetContainerRecipePolicy
	imagebuilder:GetImagePolicy
	imagebuilder:GetImageRecipePolicy
	imagebuilder:GetLifecycleExecution
	imagebuilder:GetLifecyclePolicy
	imagebuilder:GetMarketplaceResource
	imagebuilder:GetWorkflowExecution
	imagebuilder:GetWorkflowStepExecution
	imagebuilder:ImportComponent
	imagebuilder:ImportDiskImage
	imagebuilder:ImportVmImage
	imagebuilder:ListComponentBuildVersions
	imagebuilder:ListComponents
	imagebuilder:ListContainerRecipes
	imagebuilder:ListDistributionConfigurations
	imagebuilder:ListImageBuildVersions
	imagebuilder:ListImagePackages
	imagebuilder:ListImagePipelineImages
	imagebuilder:ListImagePipelines
	imagebuilder:ListImageRecipes

Prefixo do serviço	Ações do
	imagebuilder:ListImageScanFindingAggregations
	imagebuilder:ListImageScanFindings
	imagebuilder:ListImages
	imagebuilder:ListInfrastructureConfigurations
	imagebuilder:ListLifecycleExecutionResources
	imagebuilder:ListLifecycleExecutions
	imagebuilder:ListLifecyclePolicies
	imagebuilder:ListWaitingWorkflowSteps
	imagebuilder:ListWorkflowExecutions
	imagebuilder:ListWorkflowStepExecutions
	imagebuilder:ListWorkflows
	imagebuilder:PutComponentPolicy
	imagebuilder:PutContainerRecipePolicy
	imagebuilder:PutImagePolicy
	imagebuilder:PutImageRecipePolicy
	imagebuilder:RetryImage
	imagebuilder:SendWorkflowStepAction
	imagebuilder:StartImagePipelineExecution
	imagebuilder:StartResourceStateUpdate
	imagebuilder:UpdateDistributionConfiguration
	imagebuilder:UpdateImagePipeline

Prefixo do serviço	Ações do
	imagebuilder:UpdateInfrastructureConfiguration

Prefixo do serviço	Ações do
inspector	inspector:AddAttributesToFindings inspector:CreateAssessmentTarget inspector:CreateAssessmentTemplate inspector:CreateExclusionsPreview inspector:CreateResourceGroup inspector:DeleteAssessmentRun inspector:DeleteAssessmentTarget inspector:DeleteAssessmentTemplate inspector:DescribeAssessmentRuns inspector:DescribeAssessmentTargets inspector:DescribeAssessmentTemplates inspector:DescribeCrossAccountAccessRole inspector:DescribeExclusions inspector:DescribeFindings inspector:DescribeResourceGroups inspector:DescribeRulesPackages inspector:GetAssessmentReport inspector:GetExclusionsPreview inspector:GetTelemetryMetadata inspector:ListAssessmentRunAgents inspector:ListAssessmentRuns

Prefixo do serviço	Ações do
	inspector:ListAssessmentTargets inspector:ListAssessmentTemplates inspector:ListEventSubscriptions inspector:ListExclusions inspector:ListFindings inspector:ListRulesPackages inspector:PreviewAgents inspector:RegisterCrossAccountAccessRole inspector:RemoveAttributesFromFindings inspector:StartAssessmentRun inspector:StopAssessmentRun inspector:SubscribeToEvent inspector:UnsubscribeFromEvent inspector:UpdateAssessmentTarget

Prefixo do serviço	Ações do
inspector2	inspector2:AssociateMember
	inspector2:BatchGetAccountStatus
	inspector2:BatchGetCodeSnippet
	inspector2:BatchGetFindingDetails
	inspector2:BatchGetFreeTrialInfo
	inspector2:BatchGetMemberEc2DeepInspectionStatus
	inspector2:BatchUpdateMemberEc2DeepInspectionStatus
	inspector2:CancelFindingsReport
	inspector2:CancelSbomExport
	inspector2:CreateCisScanConfiguration
	inspector2:CreateCodeSecurityIntegration
	inspector2:CreateConnector
	inspector2:CreateFilter
	inspector2:CreateFindingsReport
	inspector2:CreateSbomExport
	inspector2:DeleteCisScanConfiguration
	inspector2:DeleteCodeSecurityIntegration
	inspector2:DeleteConnector
	inspector2:DeleteFilter
	inspector2:DescribeOrganizationConfiguration
	inspector2:Disable

Prefixo do serviço	Ações do
	inspector2:DisableDelegatedAdminAccount
	inspector2:DisassociateMember
	inspector2:Enable
	inspector2:EnableDelegatedAdminAccount
	inspector2:GetCisScanReport
	inspector2:GetCisScanResultDetails
	inspector2:GetClustersForImage
	inspector2:GetCodeSecurityIntegration
	inspector2:GetCodeSecurityScan
	inspector2:GetConfiguration
	inspector2:GetDelegatedAdminAccount
	inspector2:GetEc2DeepInspectionConfiguration
	inspector2:GetEncryptionKey
	inspector2:GetFindingsReportStatus
	inspector2:GetMember
	inspector2:GetSbomExport
	inspector2:ListAccountPermissions
	inspector2:ListCisScanConfigurations
	inspector2:ListCisScanResultsAggregatedByChecks
	inspector2:ListCisScanResultsAggregatedByTargetResource
	inspector2:ListCisScans

Prefixo do serviço	Ações do
	inspector2:ListCodeSecurityIntegrations
	inspector2:ListCodeSecurityScanConfigurations
	inspector2:ListConnectorScanConfigurations
	inspector2:ListConnectors
	inspector2:ListCoverage
	inspector2:ListCoverageStatistics
	inspector2:ListDelegatedAdminAccounts
	inspector2:ListFilters
	inspector2:ListFindingAggregations
	inspector2:ListFindings
	inspector2:ListMembers
	inspector2:ListUsageTotals
	inspector2:ResetEncryptionKey
	inspector2:SearchVulnerabilities
	inspector2:SendCisSessionHealth
	inspector2:SendCisSessionTelemetry
	inspector2:StartCisSession
	inspector2:StartCodeSecurityScan
	inspector2:StopCisSession
	inspector2:UpdateCisScanConfiguration
	inspector2:UpdateCodeSecurityIntegration

Prefixo do serviço	Ações do
	inspector2:UpdateConfiguration
	inspector2:UpdateConnector
	inspector2:UpdateEc2DeepInspectionConfiguration
	inspector2:UpdateEncryptionKey
	inspector2:UpdateFilter
	inspector2:UpdateOrgEc2DeepInspectionConfiguration
	inspector2:UpdateOrganizationConfiguration

Prefixo do serviço	Ações do
iot	iot:AcceptCertificateTransfer iot:AddThingToBillingGroup iot:AddThingToThingGroup iot:AssociateSbomWithPackageVersion iot:AssociateTargetsWithJob iot:AttachPolicy iot:AttachPrincipalPolicy iot:AttachSecurityProfile iot:AttachThingPrincipal iot:CancelAuditMitigationActionsTask iot:CancelAuditTask iot:CancelCertificateTransfer iot:CancelDetectMitigationActionsTask iot:CancelJob iot:CancelJobExecution iot:ClearDefaultAuthorizer iot:ConfirmTopicRuleDestination iot:CreateAuditSuppression iot:CreateAuthorizer iot:CreateBillingGroup iot:CreateCertificateFromCsr

Prefixo do serviço	Ações do
	iot:CreateCertificateProvider iot:CreateCommand iot:CreateCustomMetric iot:CreateDimension iot:CreateDomainConfiguration iot:CreateDynamicThingGroup iot:CreateFleetMetric iot:CreateJob iot:CreateJobTemplate iot:CreateKeysAndCertificate iot:CreateMitigationAction iot:CreateOTAUpdate iot:CreatePackage iot:CreatePackageVersion iot:CreatePolicy iot:CreatePolicyVersion iot:CreateProvisioningClaim iot:CreateProvisioningTemplate iot:CreateProvisioningTemplateVersion iot:CreateRoleAlias iot:CreateScheduledAudit

Prefixo do serviço	Ações do
	iot:CreateSecurityProfile iot:CreateStream iot:CreateThing iot:CreateThingGroup iot:CreateThingType iot:CreateTopicRule iot:CreateTopicRuleDestination iot:DeleteAccountAuditConfiguration iot:DeleteAuditSuppression iot:DeleteAuthorizer iot:DeleteBillingGroup iot:DeleteCACertificate iot:DeleteCertificate iot:DeleteCertificateProvider iot:DeleteCommand iot:DeleteCustomMetric iot:DeleteDimension iot:DeleteDomainConfiguration iot:DeleteDynamicThingGroup iot:DeleteFleetMetric iot:DeleteJob

Prefixo do serviço	Ações do
	iot:DeleteJobExecution iot:DeleteJobTemplate iot:DeleteMitigationAction iot:DeleteOTAUpdate iot:DeletePackage iot:DeletePackageVersion iot:DeletePolicy iot:DeletePolicyVersion iot:DeleteProvisioningTemplate iot:DeleteProvisioningTemplateVersion iot:DeleteRegistrationCode iot:DeleteRoleAlias iot:DeleteScheduledAudit iot:DeleteSecurityProfile iot:DeleteStream iot:DeleteThing iot:DeleteThingGroup iot:DeleteThingType iot:DeleteTopicRule iot:DeleteTopicRuleDestination iot:DeleteV2LoggingLevel

Prefixo do serviço	Ações do
	iot:DeprecateThingType iot:DescribeAccountAuditConfiguration iot:DescribeAuditFinding iot:DescribeAuditMitigationActionsTask iot:DescribeAuditSuppression iot:DescribeAuditTask iot:DescribeAuthorizer iot:DescribeBillingGroup iot:DescribeCACertificate iot:DescribeCertificate iot:DescribeCertificateProvider iot:DescribeCustomMetric iot:DescribeDefaultAuthorizer iot:DescribeDetectMitigationActionsTask iot:DescribeDimension iot:DescribeDomainConfiguration iot:DescribeEncryptionConfiguration iot:DescribeEndpoint iot:DescribeEventConfigurations iot:DescribeFleetMetric iot:DescribeIndex

Prefixo do serviço	Ações do
	iot:DescribeJob iot:DescribeJobExecution iot:DescribeJobTemplate iot:DescribeManagedJobTemplate iot:DescribeMitigationAction iot:DescribeProvisioningTemplate iot:DescribeProvisioningTemplateVersion iot:DescribeRoleAlias iot:DescribeScheduledAudit iot:DescribeSecurityProfile iot:DescribeStream iot:DescribeThing iot:DescribeThingGroup iot:DescribeThingRegistrationTask iot:DescribeThingType iot:DetachPolicy iot:DetachPrincipalPolicy iot:DetachSecurityProfile iot:DetachThingPrincipal iot:DisableTopicRule iot:DisassociateSbomFromPackageVersion

Prefixo do serviço	Ações do
	iot:EnableTopicRule iot:GetBehaviorModelTrainingSummaries iot:GetBucketsAggregation iot:GetCardinality iot:GetCommand iot:GetEffectivePolicies iot:GetJobDocument iot:GetLoggingOptions iot:GetOTAUpdate iot:GetPackage iot:GetPackageConfiguration iot:GetPackageVersion iot:GetPercentiles iot:GetPolicy iot:GetPolicyVersion iot:GetRegistrationCode iot:GetStatistics iot:GetThingConnectivityData iot:GetTopicRule iot:GetTopicRuleDestination iot:GetV2LoggingOptions

Prefixo do serviço	Ações do
	iot:ListActiveViolations iot:ListAttachedPolicies iot:ListAuditFindings iot:ListAuditMitigationActionsExecutions iot:ListAuditMitigationActionsTasks iot:ListAuditSuppressions iot:ListAuditTasks iot:ListAuthorizers iot:ListBillingGroups iot:ListCACertificates iot:ListCertificateProviders iot:ListCertificates iot:ListCertificatesByCA iot:ListCommands iot:ListCustomMetrics iot:ListDetectMitigationActionsExecutions iot:ListDetectMitigationActionsTasks iot:ListDimensions iot:ListDomainConfigurations iot:ListFleetMetrics iot:ListIndices

Prefixo do serviço	Ações do
	iot:ListJobExecutionsForJob iot:ListJobExecutionsForThing iot:ListJobTemplates iot:ListJobs iot:ListManagedJobTemplates iot:ListMetricValues iot:ListMitigationActions iot:ListOTAUpdates iot:ListOutgoingCertificates iot:ListPackageVersions iot:ListPackages iot:ListPolicies iot:ListPolicyPrincipals iot:ListPolicyVersions iot:ListPrincipalPolicies iot:ListPrincipalThings iot:ListPrincipalThingsV2 iot:ListProvisioningTemplateVersions iot:ListProvisioningTemplates iot:ListRelatedResourcesForAuditFinding iot:ListRoleAliases

Prefixo do serviço	Ações do
	iot:ListSbomValidationResults iot:ListScheduledAudits iot:ListSecurityProfiles iot:ListSecurityProfilesForTarget iot:ListStreams iot:ListTargetsForPolicy iot:ListTargetsForSecurityProfile iot:ListThingGroups iot:ListThingGroupsForThing iot:ListThingPrincipals iot:ListThingPrincipalsV2 iot:ListThingRegistrationTaskReports iot:ListThingRegistrationTasks iot:ListThingTypes iot:ListThings iot:ListThingsInBillingGroup iot:ListThingsInThingGroup iot:ListTopicRuleDestinations iot:ListTopicRules iot:ListV2LoggingLevels iot:ListViolationEvents

Prefixo do serviço	Ações do
	iot:PutVerificationStateOnViolation iot:RegisterCACertificate iot:RegisterCertificate iot:RegisterCertificateWithoutCA iot:RegisterThing iot:RejectCertificateTransfer iot:RemoveThingFromBillingGroup iot:RemoveThingFromThingGroup iot:ReplaceTopicRule iot:SearchIndex iot:SetDefaultAuthorizer iot:SetDefaultPolicyVersion iot:SetLoggingOptions iot:SetV2LoggingLevel iot:SetV2LoggingOptions iot:StartAuditMitigationActionsTask iot:StartDetectMitigationActionsTask iot:StartOnDemandAuditTask iot:StartThingRegistrationTask iot:StopThingRegistrationTask iot:TestAuthorization

Prefixo do serviço	Ações do
	iot:TestInvokeAuthorizer iot:TransferCertificate iot:UpdateAccountAuditConfiguration iot:UpdateAuditSuppression iot:UpdateAuthorizer iot:UpdateBillingGroup iot:UpdateCACertificate iot:UpdateCertificate iot:UpdateCertificateProvider iot:UpdateCommand iot:UpdateCustomMetric iot:UpdateDimension iot:UpdateDomainConfiguration iot:UpdateDynamicThingGroup iot:UpdateEncryptionConfiguration iot:UpdateEventConfigurations iot:UpdateFleetMetric iot:UpdateIndexingConfiguration iot:UpdateJob iot:UpdateMitigationAction iot:UpdatePackage

Prefixo do serviço	Ações do
	iot:UpdatePackageConfiguration iot:UpdatePackageVersion iot:UpdateProvisioningTemplate iot:UpdateRoleAlias iot:UpdateScheduledAudit iot:UpdateSecurityProfile iot:UpdateStream iot:UpdateThing iot:UpdateThingGroup iot:UpdateThingGroupsForThing iot:UpdateThingType iot:UpdateTopicRuleDestination iot:ValidateSecurityProfileBehaviors

Prefixo do serviço	Ações do
iotanalytics	iotanalytics:CancelPipelineReprocessing
	iotanalytics:CreateChannel
	iotanalytics:CreateDataset
	iotanalytics:CreateDatasetContent
	iotanalytics:CreateDatastore
	iotanalytics:CreatePipeline
	iotanalytics:DeleteChannel
	iotanalytics:DeleteDataset
	iotanalytics:DeleteDatasetContent
	iotanalytics:DeleteDatastore
	iotanalytics:DeletePipeline
	iotanalytics:DescribeChannel
	iotanalytics:DescribeDataset
	iotanalytics:DescribeDatastore
	iotanalytics:DescribeLoggingOptions
	iotanalytics:DescribePipeline
	iotanalytics:GetDatasetContent
	iotanalytics:ListChannels
	iotanalytics:ListDatasetContents
	iotanalytics:ListDatasets
	iotanalytics:ListDatastores

Prefixo do serviço	Ações do
	iotanalytics:ListPipelines iotanalytics:PutLoggingOptions iotanalytics:RunPipelineActivity iotanalytics:SampleChannelData iotanalytics:StartPipelineReprocessing iotanalytics:UpdateChannel iotanalytics:UpdateDataset iotanalytics:UpdateDatastore iotanalytics:UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor:CreateSuiteDefinition iotdeviceadvisor:DeleteSuiteDefinition iotdeviceadvisor:GetEndpoint iotdeviceadvisor:GetSuiteDefinition iotdeviceadvisor:GetSuiteRun iotdeviceadvisor:GetSuiteRunReport iotdeviceadvisor:ListSuiteDefinitions iotdeviceadvisor:ListSuiteRuns iotdeviceadvisor:StartSuiteRun iotdeviceadvisor:StopSuiteRun iotdeviceadvisor:UpdateSuiteDefinition

Prefixo do serviço	Ações do
iotevents	iotevents:BatchAcknowledgeAlarm
	iotevents:BatchDeleteDetector
	iotevents:BatchDisableAlarm
	iotevents:BatchEnableAlarm
	iotevents:BatchResetAlarm
	iotevents:BatchSnoozeAlarm
	iotevents:BatchUpdateDetector
	iotevents:CreateAlarmModel
	iotevents:CreateDetectorModel
	iotevents:CreateInput
	iotevents:DeleteAlarmModel
	iotevents:DeleteDetectorModel
	iotevents:DeleteInput
	iotevents:DescribeAlarm
	iotevents:DescribeAlarmModel
	iotevents:DescribeDetector
	iotevents:DescribeDetectorModel
	iotevents:DescribeDetectorModelAnalysis
	iotevents:DescribeInput
	iotevents:DescribeLoggingOptions
	iotevents:GetDetectorModelAnalysisResults

Prefixo do serviço	Ações do
	iotevents:ListAlarmModelVersions iotevents:ListAlarmModels iotevents:ListAlarms iotevents:ListDetectorModelVersions iotevents:ListDetectorModels iotevents:ListDetectors iotevents:ListInputRoutings iotevents:ListInputs iotevents:PutLoggingOptions iotevents:StartDetectorModelAnalysis iotevents:UpdateAlarmModel iotevents:UpdateDetectorModel iotevents:UpdateInput
iotfleethub	iotfleethub:CreateApplication iotfleethub:DeleteApplication iotfleethub:DescribeApplication iotfleethub:ListApplications iotfleethub:UpdateApplication

Prefixo do serviço	Ações do
iotsitewise	iotsitewise:AssociateAssets
	iotsitewise:AssociateTimeSeriesToAssetProperty
	iotsitewise:BatchAssociateProjectAssets
	iotsitewise:BatchDisassociateProjectAssets
	iotsitewise:CreateAccessPolicy
	iotsitewise:CreateAsset
	iotsitewise:CreateAssetModel
	iotsitewise:CreateAssetModelCompositeModel
	iotsitewise:CreateBulkImportJob
	iotsitewise:CreateComputationModel
	iotsitewise:CreateDashboard
	iotsitewise:CreateDataset
	iotsitewise:CreateGateway
	iotsitewise:CreatePortal
	iotsitewise:CreateProject
	iotsitewise:DeleteAccessPolicy
	iotsitewise:DeleteAsset
	iotsitewise:DeleteAssetModel
	iotsitewise:DeleteAssetModelCompositeModel
	iotsitewise:DeleteComputationModel
	iotsitewise:DeleteDashboard

Prefixo do serviço	Ações do
	iotsitewise:DeleteDataset
	iotsitewise:DeleteGateway
	iotsitewise:DeletePortal
	iotsitewise:DeleteProject
	iotsitewise:DeleteTimeSeries
	iotsitewise:DescribeAccessPolicy
	iotsitewise:DescribeAsset
	iotsitewise:DescribeAssetCompositeModel
	iotsitewise:DescribeAssetModel
	iotsitewise:DescribeAssetModelCompositeModel
	iotsitewise:DescribeAssetModelInterfaceRelationship
	iotsitewise:DescribeAssetProperty
	iotsitewise:DescribeBulkImportJob
	iotsitewise:DescribeComputationModel
	iotsitewise:DescribeComputationModelExecutionSummary
	iotsitewise:DescribeDashboard
	iotsitewise:DescribeDataset
	iotsitewise:DescribeDefaultEncryptionConfiguration
	iotsitewise:DescribeExecution
	iotsitewise:DescribeGateway
	iotsitewise:DescribeGatewayCapabilityConfiguration

Prefixo do serviço	Ações do
	iotsitewise:DescribeLoggingOptions
	iotsitewise:DescribePortal
	iotsitewise:DescribeProject
	iotsitewise:DescribeStorageConfiguration
	iotsitewise:DescribeTimeSeries
	iotsitewise:DisassociateAssets
	iotsitewise:DisassociateTimeSeriesFromAssetProperty
	iotsitewise:ExecuteAction
	iotsitewise:ExecuteQuery
	iotsitewise:ListAccessPolicies
	iotsitewise:ListActions
	iotsitewise:ListAssetModelCompositeModels
	iotsitewise:ListAssetModelProperties
	iotsitewise:ListAssetModels
	iotsitewise:ListAssetProperties
	iotsitewise:ListAssetRelationships
	iotsitewise:ListAssets
	iotsitewise:ListAssociatedAssets
	iotsitewise:ListBulkImportJobs
	iotsitewise:ListCompositionRelationships
	iotsitewise:ListComputationModelDataBindingUsages

Prefixo do serviço	Ações do
	iotsitewise:ListComputationModelResolveToResources
	iotsitewise:ListComputationModels
	iotsitewise:ListDashboards
	iotsitewise:ListDatasets
	iotsitewise:ListExecutions
	iotsitewise:ListGateways
	iotsitewise:ListInterfaceRelationships
	iotsitewise:ListPortals
	iotsitewise:ListProjectAssets
	iotsitewise:ListProjects
	iotsitewise:ListTimeSeries
	iotsitewise:PutDefaultEncryptionConfiguration
	iotsitewise:PutLoggingOptions
	iotsitewise:PutStorageConfiguration
	iotsitewise:UpdateAccessPolicy
	iotsitewise:UpdateAsset
	iotsitewise:UpdateAssetModel
	iotsitewise:UpdateAssetModelCompositeModel
	iotsitewise:UpdateAssetProperty
	iotsitewise:UpdateComputationModel
	iotsitewise:UpdateDashboard

Prefixo do serviço	Ações do
	iotsitewise:UpdateDataset
	iotsitewise:UpdateGateway
	iotsitewise:UpdateGatewayCapabilityConfiguration
	iotsitewise:UpdatePortal
	iotsitewise:UpdateProject

Prefixo do serviço	Ações do
iottwinmaker	iottwinmaker:CancelMetadataTransferJob iottwinmaker:CreateComponentType iottwinmaker:CreateEntity iottwinmaker:CreateMetadataTransferJob iottwinmaker:CreateScene iottwinmaker:CreateSyncJob iottwinmaker:CreateWorkspace iottwinmaker:DeleteComponentType iottwinmaker:DeleteEntity iottwinmaker:DeleteScene iottwinmaker:DeleteSyncJob iottwinmaker:DeleteWorkspace iottwinmaker:ExecuteQuery iottwinmaker:GetMetadataTransferJob iottwinmaker:GetPricingPlan iottwinmaker:GetScene iottwinmaker:GetSyncJob iottwinmaker:ListComponentTypes iottwinmaker:ListComponents iottwinmaker:ListEntities iottwinmaker:ListMetadataTransferJobs

Prefixo do serviço	Ações do
	iottwinmaker:ListProperties iottwinmaker:ListScenes iottwinmaker:ListSyncJobs iottwinmaker:ListSyncResources iottwinmaker:ListWorkspaces iottwinmaker:UpdateComponentType iottwinmaker:UpdateEntity iottwinmaker:UpdatePricingPlan iottwinmaker:UpdateScene iottwinmaker:UpdateWorkspace

Prefixo do serviço	Ações do
iotwireless	iotwireless:AssociateAwsAccountWithPartnerAccount
	iotwireless:AssociateMulticastGroupWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithMulticastGroup
	iotwireless:AssociateWirelessDeviceWithThing
	iotwireless:AssociateWirelessGatewayWithCertificate
	iotwireless:AssociateWirelessGatewayWithThing
	iotwireless:CancelMulticastGroupSession
	iotwireless:CreateDestination
	iotwireless:CreateDeviceProfile
	iotwireless:CreateFuotaTask
	iotwireless:CreateMulticastGroup
	iotwireless:CreateNetworkAnalyzerConfiguration
	iotwireless:CreateServiceProfile
	iotwireless:CreateWirelessDevice
	iotwireless:CreateWirelessGateway
	iotwireless:CreateWirelessGatewayTask
	iotwireless:CreateWirelessGatewayTaskDefinition
	iotwireless:DeleteDestination
	iotwireless:DeleteDeviceProfile
	iotwireless:DeleteFuotaTask

Prefixo do serviço	Ações do
	iotwireless:DeleteMulticastGroup
	iotwireless:DeleteNetworkAnalyzerConfiguration
	iotwireless:DeleteQueuedMessages
	iotwireless:DeleteServiceProfile
	iotwireless:DeleteWirelessDevice
	iotwireless:DeleteWirelessDeviceImportTask
	iotwireless:DeleteWirelessGateway
	iotwireless:DeleteWirelessGatewayTask
	iotwireless:DeleteWirelessGatewayTaskDefinition
	iotwireless:DeregisterWirelessDevice
	iotwireless:DisassociateAwsAccountFromPartnerAccount
	iotwireless:DisassociateMulticastGroupFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromMulticastGroup
	iotwireless:DisassociateWirelessDeviceFromThing
	iotwireless:DisassociateWirelessGatewayFromCertificate
	iotwireless:DisassociateWirelessGatewayFromThing
	iotwireless:GetDestination
	iotwireless:GetDeviceProfile
	iotwireless:GetEventConfigurationByResourceTypes
	iotwireless:GetFuotaTask

Prefixo do serviço	Ações do
	iotwireless:GetLogLevelsByResourceTypes
	iotwireless:GetMetricConfiguration
	iotwireless:GetMetrics
	iotwireless:GetMulticastGroup
	iotwireless:GetMulticastGroupSession
	iotwireless:GetNetworkAnalyzerConfiguration
	iotwireless:GetPartnerAccount
	iotwireless:GetPosition
	iotwireless:GetPositionConfiguration
	iotwireless:GetPositionEstimate
	iotwireless:GetResourceEventConfiguration
	iotwireless:GetResourceLogLevel
	iotwireless:GetResourcePosition
	iotwireless:GetServiceEndpoint
	iotwireless:GetServiceProfile
	iotwireless:GetWirelessDevice
	iotwireless:GetWirelessDeviceImportTask
	iotwireless:GetWirelessDeviceStatistics
	iotwireless:GetWirelessGateway
	iotwireless:GetWirelessGatewayCertificate
	iotwireless:GetWirelessGatewayFirmwareInformation

Prefixo do serviço	Ações do
	iotwireless:GetWirelessGatewayStatistics
	iotwireless:GetWirelessGatewayTask
	iotwireless:GetWirelessGatewayTaskDefinition
	iotwireless:ListDestinations
	iotwireless:ListDeviceProfiles
	iotwireless:ListDevicesForWirelessDeviceImportTask
	iotwireless:ListEventConfigurations
	iotwireless:ListFuotaTasks
	iotwireless:ListMulticastGroups
	iotwireless:ListMulticastGroupsByFuotaTask
	iotwireless:ListNetworkAnalyzerConfigurations
	iotwireless:ListPartnerAccounts
	iotwireless:ListPositionConfigurations
	iotwireless:ListQueuedMessages
	iotwireless:ListServiceProfiles
	iotwireless:ListWirelessDeviceImportTasks
	iotwireless:ListWirelessDevices
	iotwireless:ListWirelessGatewayTaskDefinitions
	iotwireless:ListWirelessGateways
	iotwireless:PutPositionConfiguration
	iotwireless:PutResourceLogLevel

Prefixo do serviço	Ações do
	iotwireless:ResetAllResourceLogLevels
	iotwireless:ResetResourceLogLevel
	iotwireless:SendDataToMulticastGroup
	iotwireless:SendDataToWirelessDevice
	iotwireless:StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless:StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless:StartFuotaTask
	iotwireless:StartMulticastGroupSession
	iotwireless:StartNetworkAnalyzerStream
	iotwireless:StartSingleWirelessDeviceImportTask
	iotwireless:StartWirelessDeviceImportTask
	iotwireless:TestWirelessDevice
	iotwireless:UpdateDestination
	iotwireless:UpdateEventConfigurationByResourceTypes
	iotwireless:UpdateFuotaTask
	iotwireless:UpdateLogLevelsByResourceTypes
	iotwireless:UpdateMetricConfiguration
	iotwireless:UpdateMulticastGroup
	iotwireless:UpdateNetworkAnalyzerConfiguration
	iotwireless:UpdatePartnerAccount

Prefixo do serviço	Ações do
	iotwireless:UpdatePosition iotwireless:UpdateResourceEventConfiguration iotwireless:UpdateResourcePosition iotwireless:UpdateWirelessDevice iotwireless:UpdateWirelessDeviceImportTask iotwireless:UpdateWirelessGateway

Prefixo do serviço	Ações do
ivs	ivs:BatchGetChannel ivs:BatchGetStreamKey ivs:BatchStartViewerSessionRevocation ivs:CreateAdConfiguration ivs:CreateChannel ivs:CreateEncoderConfiguration ivs:CreateIngestConfiguration ivs:CreateParticipantToken ivs:CreatePlaybackRestrictionPolicy ivs:CreateRecordingConfiguration ivs:CreateStorageConfiguration ivs:CreateStreamKey ivs:DeleteAdConfiguration ivs:DeleteChannel ivs:DeleteEncoderConfiguration ivs:DeleteIngestConfiguration ivs:DeletePlaybackKeyPair ivs:DeletePlaybackRestrictionPolicy ivs:DeletePublicKey ivs:DeleteRecordingConfiguration ivs:DeleteStorageConfiguration

Prefixo do serviço	Ações do
	ivs:DeleteStreamKey ivs:DisconnectParticipant ivs:GetAdConfiguration ivs:GetChannel ivs:GetComposition ivs:GetEncoderConfiguration ivs:GetIngestConfiguration ivs:GetParticipant ivs:GetPlaybackKeyPair ivs:GetPlaybackRestrictionPolicy ivs:GetPublicKey ivs:GetRecordingConfiguration ivs:GetStorageConfiguration ivs:GetStream ivs:GetStreamKey ivs:GetStreamSession ivs:ImportPlaybackKeyPair ivs:ImportPublicKey ivs:InsertAdBreak ivs:ListAdConfigurations ivs:ListChannels

Prefixo do serviço	Ações do
	ivs:ListCompositions ivs:ListEncoderConfigurations ivs:ListIngestConfigurations ivs:ListParticipantEvents ivs:ListParticipantReplicas ivs:ListParticipants ivs:ListPlaybackKeyPairs ivs:ListPlaybackRestrictionPolicies ivs:ListPublicKeys ivs:ListRecordingConfigurations ivs:ListStorageConfigurations ivs:ListStreamKeys ivs:ListStreamSessions ivs:ListStreams ivs:PutMetadata ivs:StartComposition ivs:StartViewerSessionRevocation ivs:StopComposition ivs:StopStream ivs:UpdateAdConfiguration ivs:UpdateChannel

Prefixo do serviço	Ações do
	ivs:UpdateIngestConfiguration ivs:UpdatePlaybackRestrictionPolicy
ivschat	ivschat:CreateChatToken ivschat:CreateLoggingConfiguration ivschat:CreateRoom ivschat:DeleteLoggingConfiguration ivschat:DeleteMessage ivschat:DeleteRoom ivschat:DisconnectUser ivschat:GetLoggingConfiguration ivschat:GetRoom ivschat:ListLoggingConfigurations ivschat:ListRooms ivschat:SendEvent ivschat:UpdateLoggingConfiguration ivschat:UpdateRoom

Prefixo do serviço	Ações do
kafka	kafka:BatchAssociateScramSecret kafka:BatchDisassociateScramSecret kafka:CreateCluster kafka:CreateClusterV2 kafka:CreateConfiguration kafka:CreateReplicator kafka:CreateVpcConnection kafka:DeleteCluster kafka:DeleteClusterPolicy kafka:DeleteConfiguration kafka:DeleteReplicator kafka:DeleteVpcConnection kafka:DescribeCluster kafka:DescribeClusterOperation kafka:DescribeClusterOperationV2 kafka:DescribeClusterV2 kafka:DescribeConfiguration kafka:DescribeConfigurationRevision kafka:DescribeVpcConnection kafka:GetBootstrapBrokers kafka:GetClusterPolicy

Prefixo do serviço	Ações do
	kafka:GetCompatibleKafkaVersions kafka:ListClientVpcConnections kafka:ListClusterOperations kafka:ListClusterOperationsV2 kafka:ListClusters kafka:ListClustersV2 kafka:ListConfigurationRevisions kafka:ListConfigurations kafka:ListKafkaVersions kafka:ListNodes kafka:ListReplicators kafka:ListScramSecrets kafka:ListVpcConnections kafka:PutClusterPolicy kafka:RebootBroker kafka:RejectClientVpcConnection kafka:UpdateBrokerCount kafka:UpdateBrokerStorage kafka:UpdateBrokerType kafka:UpdateClusterConfiguration kafka:UpdateClusterKafkaVersion

Prefixo do serviço	Ações do
	kafka:UpdateConfiguration kafka:UpdateConnectivity kafka:UpdateMonitoring kafka:UpdateRebalancing kafka:UpdateReplicationInfo kafka:UpdateSecurity kafka:UpdateStorage
kafkaconnect	kafkaconnect:CreateConnector kafkaconnect:CreateCustomPlugin kafkaconnect:CreateWorkerConfiguration kafkaconnect:DeleteConnector kafkaconnect:DeleteCustomPlugin kafkaconnect:DeleteWorkerConfiguration kafkaconnect:DescribeConnector kafkaconnect:DescribeCustomPlugin kafkaconnect:DescribeWorkerConfiguration kafkaconnect:ListConnectorOperations kafkaconnect:ListConnectors kafkaconnect:ListCustomPlugins kafkaconnect:ListWorkerConfigurations kafkaconnect:UpdateConnector

Prefixo do serviço	Ações do
kendra	kendra:AssociateEntitiesToExperience kendra:AssociatePersonasToEntities kendra:BatchDeleteDocument kendra:BatchDeleteFeaturedResultsSet kendra:BatchGetDocumentStatus kendra:BatchPutDocument kendra:ClearQuerySuggestions kendra:CreateAccessControlConfiguration kendra:CreateDataSource kendra:CreateExperience kendra:CreateFaq kendra:CreateFeaturedResultsSet kendra:CreateIndex kendra:CreateQuerySuggestionsBlockList kendra:CreateThesaurus kendra:DeleteDataSource kendra:DeleteExperience kendra:DeleteFaq kendra:DeleteIndex kendra:DeletePrincipalMapping kendra:DeleteQuerySuggestionsBlockList

Prefixo do serviço	Ações do
	kendra:DeleteThesaurus kendra:DescribeAccessControlConfiguration kendra:DescribeDataSource kendra:DescribeExperience kendra:DescribeFaq kendra:DescribeFeaturedResultsSet kendra:DescribeIndex kendra:DescribePrincipalMapping kendra:DescribeQuerySuggestionsBlockList kendra:DescribeQuerySuggestionsConfig kendra:DescribeThesaurus kendra:DisassociateEntitiesFromExperience kendra:DisassociatePersonasFromEntities kendra:GetQuerySuggestions kendra:GetSnapshots kendra:ListAccessControlConfigurations kendra:ListDataSourceSyncJobs kendra:ListDataSources kendra:ListEntityPersonas kendra:ListExperienceEntities kendra:ListExperiences

Prefixo do serviço	Ações do
	kendra:ListFaqs kendra:ListFeaturedResultsSets kendra:ListGroupsOlderThanOrderingId kendra:ListIndices kendra:ListQuerySuggestionsBlockLists kendra:ListThesauri kendra:PutPrincipalMapping kendra:Query kendra:Retrieve kendra:StartDataSourceSyncJob kendra:StopDataSourceSyncJob kendra:SubmitFeedback kendra:UpdateDataSource kendra:UpdateExperience kendra:UpdateFeaturedResultsSet kendra:UpdateIndex kendra:UpdateQuerySuggestionsBlockList kendra:UpdateQuerySuggestionsConfig kendra:UpdateThesaurus

Prefixo do serviço	Ações do
kinesis	kinesis:CreateStream
	kinesis:DecreaseStreamRetentionPeriod
	kinesis:DeleteStream
	kinesis:DeregisterStreamConsumer
	kinesis:DescribeAccountSettings
	kinesis:DescribeLimits
	kinesis:DescribeStream
	kinesis:DescribeStreamConsumer
	kinesis:DescribeStreamSummary
	kinesis:DisableEnhancedMonitoring
	kinesis:EnableEnhancedMonitoring
	kinesis:IncreaseStreamRetentionPeriod
	kinesis:ListShards
	kinesis:ListStreamConsumers
	kinesis:ListStreams
	kinesis:MergeShards
	kinesis:RegisterStreamConsumer
	kinesis:SplitShard
	kinesis:StartStreamEncryption
	kinesis:StopStreamEncryption
	kinesis:UpdateAccountSettings

Prefixo do serviço	Ações do
	kinesis:UpdateShardCount kinesis:UpdateStreamMode

Prefixo do serviço	Ações do
kinesisanalytics	kinesisanalytics:AddApplicationCloudWatchLoggingOption kinesisanalytics:AddApplicationInput kinesisanalytics:AddApplicationInputProcessingConfiguration kinesisanalytics:AddApplicationOutput kinesisanalytics:AddApplicationReferenceDataSource kinesisanalytics:AddApplicationVpcConfiguration kinesisanalytics:CreateApplication kinesisanalytics:CreateApplicationPresignedUrl kinesisanalytics:CreateApplicationSnapshot kinesisanalytics:DeleteApplication kinesisanalytics:DeleteApplicationCloudWatchLoggingOption kinesisanalytics:DeleteApplicationInputProcessingConfiguration kinesisanalytics:DeleteApplicationOutput kinesisanalytics:DeleteApplicationReferenceDataSource kinesisanalytics:DeleteApplicationSnapshot kinesisanalytics:DeleteApplicationVpcConfiguration kinesisanalytics:DescribeApplication kinesisanalytics:DescribeApplicationOperation kinesisanalytics:DescribeApplicationSnapshot kinesisanalytics:DescribeApplicationVersion kinesisanalytics:DiscoverInputSchema

Prefixo do serviço	Ações do
	kinesisanalytics:ListApplicationOperations kinesisanalytics:ListApplicationSnapshots kinesisanalytics:ListApplicationVersions kinesisanalytics:ListApplications kinesisanalytics:RollbackApplication kinesisanalytics:StartApplication kinesisanalytics:StopApplication kinesisanalytics:UpdateApplication kinesisanalytics:UpdateApplicationMaintenanceConfiguration

Prefixo do serviço	Ações do
kms	kms:CancelKeyDeletion
	kms:ConnectCustomKeyStore
	kms:CreateAlias
	kms:CreateCustomKeyStore
	kms:CreateGrant
	kms:CreateKey
	kms:Decrypt
	kms:DeleteAlias
	kms:DeleteCustomKeyStore
	kms:DeleteImportedKeyMaterial
	kms:DeriveSharedSecret
	kms:DescribeCustomKeyStores
	kms:DescribeKey
	kms:DisableKey
	kms:DisableKeyRotation
	kms:DisconnectCustomKeyStore
	kms:EnableKey
	kms:EnableKeyRotation
	kms:Encrypt
	kms:GenerateDataKey
	kms:GenerateDataKeyPair

Prefixo do serviço	Ações do
	kms:GenerateDataKeyPairWithoutPlaintext
	kms:GenerateDataKeyWithoutPlaintext
	kms:GenerateMac
	kms:GenerateRandom
	kms:GetKeyLastUsage
	kms:GetKeyPolicy
	kms:GetKeyRotationStatus
	kms:GetParametersForImport
	kms:GetPublicKey
	kms:ImportKeyMaterial
	kms:ListAliases
	kms:ListGrants
	kms:ListKeyPolicies
	kms:ListKeyRotations
	kms:ListKeys
	kms:ListRetirableGrants
	kms:ReplicateKey
	kms:RetireGrant
	kms:RevokeGrant
	kms:RotateKeyOnDemand
	kms:ScheduleKeyDeletion

Prefixo do serviço	Ações do
	kms:Sign
	kms:UpdateAlias
	kms:UpdateCustomKeyStore
	kms:UpdateKeyDescription
	kms:UpdatePrimaryRegion
	kms:Verify
	kms:VerifyMac

Prefixo do serviço	Ações do
lambda	lambda:AddLayerVersionPermission lambda:AddPermission lambda:CreateAlias lambda:CreateCodeSigningConfig lambda:CreateEventSourceMapping lambda:CreateFunction lambda:CreateFunctionUrlConfig lambda:DeleteAlias lambda:DeleteCapacityProvider lambda:DeleteCodeSigningConfig lambda:DeleteEventSourceMapping lambda:DeleteFunction lambda:DeleteFunctionCodeSigningConfig lambda:DeleteFunctionConcurrency lambda:DeleteFunctionEventInvokeConfig lambda:DeleteFunctionUrlConfig lambda:DeleteLayerVersion lambda:DeleteProvisionedConcurrencyConfig lambda:GetAccountSettings lambda:GetAlias lambda:GetCapacityProvider

Prefixo do serviço	Ações do
	lambda:GetCodeSigningConfig lambda:GetEventSourceMapping lambda:GetFunction lambda:GetFunctionCodeSigningConfig lambda:GetFunctionConcurrency lambda:GetFunctionConfiguration lambda:GetFunctionEventInvokeConfig lambda:GetFunctionRecursionConfig lambda:GetFunctionScalingConfig lambda:GetFunctionUrlConfig lambda:GetLayerVersion lambda:GetLayerVersionPolicy lambda:GetPolicy lambda:GetProvisionedConcurrencyConfig lambda:GetRuntimeManagementConfig lambda:ListAliases lambda:ListCapacityProviders lambda:ListCodeSigningConfigs lambda:ListDurableExecutionsByFunction lambda:ListEventSourceMappings lambda:ListFunctionEventInvokeConfigs

Prefixo do serviço	Ações do
	lambda:ListFunctionUrlConfigs lambda:ListFunctions lambda:ListFunctionsByCodeSigningConfig lambda:ListLayerVersions lambda:ListLayers lambda:ListProvisionedConcurrencyConfigs lambda:ListVersionsByFunction lambda:PublishLayerVersion lambda:PublishVersion lambda:PutFunctionCodeSigningConfig lambda:PutFunctionConcurrency lambda:PutFunctionEventInvokeConfig lambda:PutFunctionRecursionConfig lambda:PutFunctionScalingConfig lambda:PutProvisionedConcurrencyConfig lambda:PutRuntimeManagementConfig lambda:RemoveLayerVersionPermission lambda:RemovePermission lambda:UpdateAlias lambda:UpdateCapacityProvider lambda:UpdateCodeSigningConfig

Prefixo do serviço	Ações do
	lambda:UpdateEventSourceMapping lambda:UpdateFunctionCode lambda:UpdateFunctionConfiguration lambda:UpdateFunctionEventInvokeConfig lambda:UpdateFunctionUrlConfig

Prefixo do serviço	Ações do
lex	lex:BatchCreateCustomVocabularyItem
	lex:BatchDeleteCustomVocabularyItem
	lex:BatchUpdateCustomVocabularyItem
	lex:BuildBotLocale
	lex:CreateBotAlias
	lex:CreateBotReplica
	lex:CreateBotVersion
	lex:CreateExport
	lex:CreateIntentVersion
	lex:CreateResourcePolicy
	lex:CreateSlotTypeVersion
	lex:CreateTestSetDiscrepancyReport
	lex:CreateUploadUrl
	lex:DeleteBot
	lex:DeleteBotAnalyzerRecommendation
	lex:DeleteBotChannelAssociation
	lex:DeleteBotReplica
	lex:DeleteExport
	lex:DeleteImport
	lex:DeleteIntentVersion
	lex:DeleteResourcePolicy

Prefixo do serviço	Ações do
	lex:DeleteSlotTypeVersion
	lex:DeleteTestSet
	lex:DeleteUtterances
	lex:DescribeBotAlias
	lex:DescribeBotAnalyzerRecommendation
	lex:DescribeBotRecommendation
	lex:DescribeBotReplica
	lex:DescribeBotResourceGeneration
	lex:DescribeBotVersion
	lex:DescribeCustomVocabularyMetadata
	lex:DescribeExport
	lex:DescribeImport
	lex:DescribeResourcePolicy
	lex:DescribeTestExecution
	lex:DescribeTestSet
	lex:DescribeTestSetDiscrepancyReport
	lex:DescribeTestSetGeneration
	lex:GenerateBotElement
	lex:GetBot
	lex:GetBotAlias
	lex:GetBotAliases

Prefixo do serviço	Ações do
	lex:GetBotChannelAssociation
	lex:GetBotChannelAssociations
	lex:GetBotVersions
	lex:GetBots
	lex:GetBuiltinIntent
	lex:GetBuiltinIntents
	lex:GetBuiltinSlotTypes
	lex:GetExport
	lex:GetImport
	lex:GetIntent
	lex:GetIntentVersions
	lex:GetIntents
	lex:GetMigration
	lex:GetMigrations
	lex:GetSlotType
	lex:GetSlotTypeVersions
	lex:GetSlotTypes
	lex:GetTestExecutionArtifactsUrl
	lex:GetUtterancesView
	lex:ListBotAliasReplicas
	lex:ListBotAliases

Prefixo do serviço	Ações do
	lex:ListBotRecommendations
	lex:ListBotReplicas
	lex:ListBotResourceGenerations
	lex:ListBotVersionReplicas
	lex:ListBotVersions
	lex:ListBots
	lex:ListBuiltInIntents
	lex:ListBuiltInSlotTypes
	lex:ListCustomVocabularyItems
	lex:ListExports
	lex:ListImports
	lex:ListIntentMetrics
	lex:ListIntentPaths
	lex:ListRecommendedIntents
	lex:ListSessionAnalyticsData
	lex:ListSessionMetrics
	lex:ListTestExecutionResultItems
	lex:ListTestExecutions
	lex:ListTestSets
	lex:PutBot
	lex:PutBotAlias

Prefixo do serviço	Ações do
	lex:PutIntent
	lex:PutSlotType
	lex:SearchAssociatedTranscripts
	lex:StartBotAnalyzer
	lex:StartBotRecommendation
	lex:StartImport
	lex:StartMigration
	lex:StartTestExecution
	lex:StartTestSetGeneration
	lex:StopBotAnalyzer
	lex:StopBotRecommendation
	lex:UpdateBotAlias
	lex:UpdateBotRecommendation
	lex:UpdateExport
	lex:UpdateResourcePolicy

Prefixo do serviço	Ações do
license-manager-linux-subscriptions	license-manager-linux-subscriptions:DeregisterSubscriptionProvider license-manager-linux-subscriptions:GetRegisteredSubscriptionProvider license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:ListRegisteredSubscriptionProviders license-manager-linux-subscriptions:RegisterSubscriptionProvider license-manager-linux-subscriptions:UpdateServiceSettings

Prefixo do serviço	Ações do
lightsail	lightsail:AllocateStaticIp
	lightsail:AttachCertificateToDistribution
	lightsail:AttachDisk
	lightsail:AttachInstancesToLoadBalancer
	lightsail:AttachLoadBalancerTlsCertificate
	lightsail:AttachStaticIp
	lightsail:CloseInstancePublicPorts
	lightsail:CopySnapshot
	lightsail:CreateBucket
	lightsail:CreateBucketAccessKey
	lightsail:CreateCertificate
	lightsail:CreateCloudFormationStack
	lightsail:CreateContactMethod
	lightsail:CreateContainerService
	lightsail:CreateContainerServiceDeployment
	lightsail:CreateContainerServiceRegistryLogin
	lightsail:CreateDisk
	lightsail:CreateDiskFromSnapshot
	lightsail:CreateDiskSnapshot
	lightsail:CreateDistribution
	lightsail:CreateDomain

Prefixo do serviço	Ações do
	lightsail:CreateGUISessionAccessDetails
	lightsail:CreateInstanceSnapshot
	lightsail:CreateInstances
	lightsail:CreateInstancesFromSnapshot
	lightsail:CreateKeyPair
	lightsail:CreateLoadBalancer
	lightsail:CreateLoadBalancerTlsCertificate
	lightsail:CreateRelationalDatabase
	lightsail:CreateRelationalDatabaseFromSnapshot
	lightsail:CreateRelationalDatabaseSnapshot
	lightsail:DeleteAlarm
	lightsail:DeleteAutoSnapshot
	lightsail:DeleteBucket
	lightsail:DeleteBucketAccessKey
	lightsail:DeleteCertificate
	lightsail:DeleteContactMethod
	lightsail:DeleteContainerImage
	lightsail:DeleteContainerService
	lightsail:DeleteDisk
	lightsail:DeleteDiskSnapshot
	lightsail:DeleteDistribution

Prefixo do serviço	Ações do
	lightsail:DeleteDomain
	lightsail:DeleteDomainEntry
	lightsail:DeleteInstance
	lightsail:DeleteInstanceSnapshot
	lightsail:DeleteKeyPair
	lightsail:DeleteKnownHostKeys
	lightsail:DeleteLoadBalancer
	lightsail:DeleteLoadBalancerTlsCertificate
	lightsail:DeleteRelationalDatabase
	lightsail:DeleteRelationalDatabaseSnapshot
	lightsail:DetachCertificateFromDistribution
	lightsail:DetachDisk
	lightsail:DetachInstancesFromLoadBalancer
	lightsail:DetachStaticIp
	lightsail:DisableAddOn
	lightsail:DownloadDefaultKeyPair
	lightsail:EnableAddOn
	lightsail:ExportSnapshot
	lightsail:GetActiveNames
	lightsail:GetAlarms
	lightsail:GetAutoSnapshots

Prefixo do serviço	Ações do
	lightsail:GetBlueprints
	lightsail:GetBucketAccessKeys
	lightsail:GetBucketBundles
	lightsail:GetBucketMetricData
	lightsail:GetBuckets
	lightsail:GetBundles
	lightsail:GetCertificates
	lightsail:GetCloudFormationStackRecords
	lightsail:GetContactMethods
	lightsail:GetContainerAPIMetadata
	lightsail:GetContainerImages
	lightsail:GetContainerLog
	lightsail:GetContainerServiceDeployments
	lightsail:GetContainerServiceMetricData
	lightsail:GetContainerServicePowers
	lightsail:GetContainerServices
	lightsail:GetCostEstimate
	lightsail:GetDisk
	lightsail:GetDiskSnapshot
	lightsail:GetDiskSnapshots
	lightsail:GetDisks

Prefixo do serviço	Ações do
	lightsail:GetDistributionBundles
	lightsail:GetDistributionLatestCacheReset
	lightsail:GetDistributionMetricData
	lightsail:GetDistributions
	lightsail:GetDomain
	lightsail:GetExportSnapshotRecords
	lightsail:GetInstance
	lightsail:GetInstanceMetricData
	lightsail:GetInstancePortStates
	lightsail:GetInstanceSnapshot
	lightsail:GetInstanceSnapshots
	lightsail:GetInstanceState
	lightsail:GetInstances
	lightsail:GetKeyPair
	lightsail:GetKeyPairs
	lightsail:GetLoadBalancer
	lightsail:GetLoadBalancerMetricData
	lightsail:GetLoadBalancerTlsCertificates
	lightsail:GetLoadBalancerTlsPolicies
	lightsail:GetLoadBalancers
	lightsail:GetOperation

Prefixo do serviço	Ações do
	lightsail:GetOperations
	lightsail:GetOperationsForResource
	lightsail:GetRegions
	lightsail:GetRelationalDatabase
	lightsail:GetRelationalDatabaseBlueprints
	lightsail:GetRelationalDatabaseBundles
	lightsail:GetRelationalDatabaseEvents
	lightsail:GetRelationalDatabaseLogEvents
	lightsail:GetRelationalDatabaseLogStreams
	lightsail:GetRelationalDatabaseMasterUserPassword
	lightsail:GetRelationalDatabaseMetricData
	lightsail:GetRelationalDatabaseParameters
	lightsail:GetRelationalDatabaseSnapshot
	lightsail:GetRelationalDatabaseSnapshots
	lightsail:GetRelationalDatabases
	lightsail:GetSetupHistory
	lightsail:GetStaticIp
	lightsail:GetStaticIps
	lightsail:ImportKeyPair
	lightsail:IsVpcPeered
	lightsail:OpenInstancePublicPorts

Prefixo do serviço	Ações do
	lightsail:PeerVpc
	lightsail:PutAlarm
	lightsail:PutInstancePublicPorts
	lightsail:RebootInstance
	lightsail:RebootRelationalDatabase
	lightsail:RegisterContainerImage
	lightsail:ReleaseStaticIp
	lightsail:ResetDistributionCache
	lightsail:SendContactMethodVerification
	lightsail:SetIpAddressType
	lightsail:SetResourceAccessForBucket
	lightsail:SetupInstanceHttps
	lightsail:StartGUISession
	lightsail:StartInstance
	lightsail:StartRelationalDatabase
	lightsail:StopGUISession
	lightsail:StopInstance
	lightsail:StopRelationalDatabase
	lightsail:TestAlarm
	lightsail:UnpeerVpc
	lightsail:UpdateBucket

Prefixo do serviço	Ações do
	lightsail:UpdateBucketBundle
	lightsail:UpdateContainerService
	lightsail:UpdateDistribution
	lightsail:UpdateDistributionBundle
	lightsail:UpdateDomainEntry
	lightsail:UpdateInstanceMetadataOptions
	lightsail:UpdateLoadBalancerAttribute
	lightsail:UpdateRelationalDatabase
	lightsail:UpdateRelationalDatabaseParameters

Prefixo do serviço	Ações do
logs	logs:AssociateKmsKey logs:AssociateSourceToS3TableIntegration logs:CancelExportTask logs:CancelImportTask logs:CreateDelivery logs:CreateExportTask logs:CreateLogAnomalyDetector logs:CreateLogGroup logs:CreateLogStream logs:CreateLookupTable logs:DeleteDataProtectionPolicy logs:DeleteDelivery logs:DeleteDeliveryDestination logs:DeleteDeliveryDestinationPolicy logs:DeleteDeliverySource logs:DeleteDestination logs:DeleteIndexPolicy logs:DeleteIntegration logs:DeleteLogAnomalyDetector logs:DeleteLogGroup logs:DeleteLogStream

Prefixo do serviço	Ações do
	logs:DeleteLookupTable logs:DeleteMetricFilter logs:DeleteQueryDefinition logs:DeleteResourcePolicy logs:DeleteRetentionPolicy logs:DeleteScheduledQuery logs:DeleteSubscriptionFilter logs:DeleteSyslogConfiguration logs:DeleteTransformer logs:DescribeAccountPolicies logs:DescribeConfigurationTemplates logs:DescribeDeliveries logs:DescribeDeliveryDestinations logs:DescribeDeliverySources logs:DescribeDestinations logs:DescribeExportTasks logs:DescribeFieldIndexes logs:DescribeImportTaskBatches logs:DescribeImportTasks logs:DescribeIndexPolicies logs:DescribeLogGroups

Prefixo do serviço	Ações do
	logs:DescribeLogStreams logs:DescribeLookupTables logs:DescribeMetricFilters logs:DescribeQueries logs:DescribeQueryDefinitions logs:DescribeResourcePolicies logs:DescribeSubscriptionFilters logs:DisassociateKmsKey logs:DisassociateSourceFromS3TableIntegration logs:GetDataProtectionPolicy logs:GetDelivery logs:GetDeliveryDestination logs:GetDeliveryDestinationPolicy logs:GetDeliverySource logs:GetIntegration logs:GetLogAnomalyDetector logs:GetLogFields logs:GetLogGroupFields logs:GetLogRecord logs:GetLookupTable logs:GetQueryResults

Prefixo do serviço	Ações do
	logs:GetScheduledQuery logs:GetScheduledQueryHistory logs:GetStorageTierPolicy logs:GetTransformer logs:ListAnomalies logs:ListIntegrations logs:ListLogAnomalyDetectors logs:ListLogGroupsForQuery logs:ListScheduledQueries logs:ListSourcesForS3TableIntegration logs:ListSyslogConfigurations logs:PutBearerTokenAuthentication logs:PutDataProtectionPolicy logs:PutDeliveryDestination logs:PutDeliveryDestinationPolicy logs:PutDeliverySource logs:PutDestination logs:PutDestinationPolicy logs:PutIndexPolicy logs:PutIntegration logs:PutLogGroupDeletionProtection

Prefixo do serviço	Ações do
	logs:PutMetricFilter logs:putQueryDefinition logs:PutResourcePolicy logs:PutRetentionPolicy logs:PutStorageTierPolicy logs:PutSubscriptionFilter logs:PutSyslogConfiguration logs:PutTransformer logs:StartLiveTail logs:StartQuery logs:StopQuery logs:TestMetricFilter logs:TestTransformer logs:UpdateAnomaly logs:UpdateDeliveryConfiguration logs:UpdateLogAnomalyDetector logs:UpdateLookupTable

Prefixo do serviço	Ações do
lookoutequipment	lookoutequipment:CreateDataset
	lookoutequipment:CreateInferenceScheduler
	lookoutequipment:CreateLabel
	lookoutequipment:CreateLabelGroup
	lookoutequipment:CreateModel
	lookoutequipment:DeleteDataset
	lookoutequipment:DeleteInferenceScheduler
	lookoutequipment:DeleteLabel
	lookoutequipment:DeleteLabelGroup
	lookoutequipment:DeleteModel
	lookoutequipment:DeleteResourcePolicy
	lookoutequipment:DeleteRetrainingScheduler
	lookoutequipment:DescribeDataIngestionJob
	lookoutequipment:DescribeDataset
	lookoutequipment:DescribeInferenceScheduler
	lookoutequipment:DescribeLabelGroup
	lookoutequipment:DescribeModel
	lookoutequipment:DescribeModelVersion
	lookoutequipment:DescribeResourcePolicy
	lookoutequipment:DescribeRetrainingScheduler
	lookoutequipment:Describelabel

Prefixo do serviço	Ações do
	lookoutequipment:ImportDataset
	lookoutequipment:ImportModelVersion
	lookoutequipment:ListDataIngestionJobs
	lookoutequipment:ListDatasets
	lookoutequipment:ListInferenceEvents
	lookoutequipment:ListInferenceExecutions
	lookoutequipment:ListInferenceSchedulers
	lookoutequipment:ListLabelGroups
	lookoutequipment:ListLabels
	lookoutequipment:ListModelVersions
	lookoutequipment:ListModels
	lookoutequipment:ListRetrainingSchedulers
	lookoutequipment:ListSensorStatistics
	lookoutequipment:PutResourcePolicy
	lookoutequipment:StartDataIngestionJob
	lookoutequipment:StartInferenceScheduler
	lookoutequipment:StartRetrainingScheduler
	lookoutequipment:StopInferenceScheduler
	lookoutequipment:StopRetrainingScheduler
	lookoutequipment:UpdateActiveModelVersion
	lookoutequipment:UpdateInferenceScheduler

Prefixo do serviço	Ações do
	lookoutequipment:UpdateLabelGroup lookoutequipment:UpdateModel lookoutequipment:UpdateRetrainingScheduler

Prefixo do serviço	Ações do
lookoutmetrics	lookoutmetrics:ActivateAnomalyDetector lookoutmetrics:BackTestAnomalyDetector lookoutmetrics:CreateAlert lookoutmetrics:CreateAnomalyDetector lookoutmetrics:CreateMetricSet lookoutmetrics:DeactivateAnomalyDetector lookoutmetrics:DeleteAlert lookoutmetrics:DeleteAnomalyDetector lookoutmetrics:DescribeAlert lookoutmetrics:DescribeAnomalyDetectionExecutions lookoutmetrics:DescribeAnomalyDetector lookoutmetrics:DescribeMetricSet lookoutmetrics:DetectMetricSetConfig lookoutmetrics:GetAnomalyGroup lookoutmetrics:GetDataQualityMetrics lookoutmetrics:GetFeedback lookoutmetrics:GetSampleData lookoutmetrics:ListAlerts lookoutmetrics:ListAnomalyDetectors lookoutmetrics:ListAnomalyGroupRelatedMetrics lookoutmetrics:ListAnomalyGroupSummaries

Prefixo do serviço	Ações do
	lookoutmetrics:ListAnomalyGroupTimeSeries lookoutmetrics:ListMetricSets lookoutmetrics:PutFeedback lookoutmetrics:UpdateAlert lookoutmetrics:UpdateAnomalyDetector lookoutmetrics:UpdateMetricSet

Prefixo do serviço	Ações do
m2	m2:CancelBatchJobExecution
	m2:CreateApplication
	m2:CreateDataSetExportTask
	m2:CreateDataSetImportTask
	m2:CreateDeployment
	m2:CreateEnvironment
	m2:DeleteApplication
	m2:DeleteApplicationFromEnvironment
	m2:DeleteEnvironment
	m2:GetApplication
	m2:GetApplicationVersion
	m2:GetBatchJobExecution
	m2:GetDataSetDetails
	m2:GetDataSetExportTask
	m2:GetDataSetImportTask
	m2:GetDeployment
	m2:GetEnvironment
	m2:GetSignedBluinsightsUrl
	m2:ListApplicationVersions
	m2:ListApplications
	m2:ListBatchJobDefinitions

Prefixo do serviço	Ações do
	m2:ListBatchJobExecutions
	m2:ListBatchJobRestartPoints
	m2:ListDataSetExportHistory
	m2:ListDataSetImportHistory
	m2:ListDataSets
	m2:ListDeployments
	m2:ListEngineVersions
	m2:ListEnvironments
	m2:StartApplication
	m2:StartBatchJob
	m2:StopApplication
	m2:UpdateApplication
	m2:UpdateEnvironment

Prefixo do serviço	Ações do
managedblockchain	managedblockchain:CreateAccessor managedblockchain:CreateMember managedblockchain:CreateNetwork managedblockchain:CreateNode managedblockchain:CreateProposal managedblockchain:DeleteAccessor managedblockchain:DeleteMember managedblockchain:DeleteNode managedblockchain:GetAccessor managedblockchain:GetMember managedblockchain:GetNetwork managedblockchain:GetNode managedblockchain:GetProposal managedblockchain:InvokeRpcPolygonMainnet managedblockchain:InvokeRpcPolygonMumbaiTestnet managedblockchain:ListAccessors managedblockchain:ListInvitations managedblockchain:ListMembers managedblockchain:ListNetworks managedblockchain:ListNodes managedblockchain:ListProposalVotes

Prefixo do serviço	Ações do
	managedblockchain:ListProposals managedblockchain:RejectInvitation managedblockchain:UpdateMember managedblockchain:UpdateNode managedblockchain:VoteOnProposal

Prefixo do serviço	Ações do
mediaconnect	mediaconnect:AddBridgeOutputs
	mediaconnect:AddBridgeSources
	mediaconnect:AddFlowMediaStreams
	mediaconnect:AddFlowOutputs
	mediaconnect:AddFlowSources
	mediaconnect:AddFlowVpcInterfaces
	mediaconnect:CreateBridge
	mediaconnect:CreateFlow
	mediaconnect:CreateGateway
	mediaconnect:DeleteBridge
	mediaconnect:DeleteFlow
	mediaconnect:DeleteGateway
	mediaconnect:DeleteRouterInput
	mediaconnect:DeleteRouterNetworkInterface
	mediaconnect:DeleteRouterOutput
	mediaconnect:DeregisterGatewayInstance
	mediaconnect:DescribeBridge
	mediaconnect:DescribeFlow
	mediaconnect:DescribeFlowSourceMetadata
	mediaconnect:DescribeFlowSourceThumbnail
	mediaconnect:DescribeGateway

Prefixo do serviço	Ações do
	mediaconnect:DescribeGatewayInstance
	mediaconnect:DescribeOffering
	mediaconnect:DescribeReservation
	mediaconnect:GetRouterInput
	mediaconnect:GetRouterInputSourceMetadata
	mediaconnect:GetRouterInputThumbnail
	mediaconnect:GetRouterNetworkInterface
	mediaconnect:GetRouterOutput
	mediaconnect:GrantFlowEntitlements
	mediaconnect:ListBridges
	mediaconnect:ListEntitlements
	mediaconnect:ListFlows
	mediaconnect:ListGatewayInstances
	mediaconnect:ListGateways
	mediaconnect:ListOfferings
	mediaconnect:ListReservations
	mediaconnect:ListRouterInputs
	mediaconnect:ListRouterNetworkInterfaces
	mediaconnect:ListRouterOutputs
	mediaconnect:PurchaseOffering
	mediaconnect:RemoveBridgeOutput

Prefixo do serviço	Ações do
	mediaconnect:RemoveBridgeSource
	mediaconnect:RemoveFlowMediaStream
	mediaconnect:RemoveFlowOutput
	mediaconnect:RemoveFlowSource
	mediaconnect:RemoveFlowVpcInterface
	mediaconnect:RestartRouterInput
	mediaconnect:RestartRouterOutput
	mediaconnect:RevokeFlowEntitlement
	mediaconnect:StartFlow
	mediaconnect:StartRouterInput
	mediaconnect:StartRouterOutput
	mediaconnect:StopFlow
	mediaconnect:StopRouterInput
	mediaconnect:StopRouterOutput
	mediaconnect:TakeRouterInput
	mediaconnect:UpdateBridge
	mediaconnect:UpdateBridgeOutput
	mediaconnect:UpdateBridgeSource
	mediaconnect:UpdateBridgeState
	mediaconnect:UpdateFlow
	mediaconnect:UpdateFlowEntitlement

Prefixo do serviço	Ações do
	mediaconnect:UpdateFlowMediaStream mediaconnect:UpdateGatewayInstance

Prefixo do serviço	Ações do
mediaconvert	mediaconvert:AssociateCertificate mediaconvert:CancelJob mediaconvert:CreateJob mediaconvert:CreateJobTemplate mediaconvert:CreatePreset mediaconvert:CreateQueue mediaconvert:CreateResourceShare mediaconvert:DeleteJobTemplate mediaconvert:DeletePolicy mediaconvert:DeletePreset mediaconvert:DeleteQueue mediaconvert:DescribeEndpoints mediaconvert:DisassociateCertificate mediaconvert:GetJob mediaconvert:GetJobTemplate mediaconvert:GetPolicy mediaconvert:GetPreset mediaconvert:GetQueue mediaconvert:ListJobTemplates mediaconvert:ListJobs mediaconvert:ListPresets

Prefixo do serviço	Ações do
	mediaconvert:ListQueues mediaconvert:ListVersions mediaconvert:Probe mediaconvert:PutPolicy mediaconvert:SearchJobs mediaconvert:UpdateJobTemplate mediaconvert:UpdatePreset mediaconvert:UpdateQueue

Prefixo do serviço	Ações do
medialive	medialive:AcceptInputDeviceTransfer
	medialive:BatchDelete
	medialive:BatchStart
	medialive:BatchStop
	medialive:BatchUpdateSchedule
	medialive:CancelInputDeviceTransfer
	medialive:ClaimDevice
	medialive:CreateChannel
	medialive:CreateChannelPlacementGroup
	medialive:CreateCloudWatchAlarmTemplate
	medialive:CreateCloudWatchAlarmTemplateGroup
	medialive:CreateCluster
	medialive:CreateEventBridgeRuleTemplate
	medialive:CreateEventBridgeRuleTemplateGroup
	medialive:CreateInput
	medialive:CreateInputSecurityGroup
	medialive:CreateMultiplex
	medialive:CreateMultiplexProgram
	medialive:CreateNetwork
	medialive:CreateNode
	medialive:CreateNodeRegistrationScript

Prefixo do serviço	Ações do
	medialive:CreatePartnerInput
	medialive:CreateSdiSource
	medialive:CreateSignalMap
	medialive:DeleteChannel
	medialive:DeleteChannelPlacementGroup
	medialive:DeleteCloudWatchAlarmTemplate
	medialive:DeleteCloudWatchAlarmTemplateGroup
	medialive:DeleteCluster
	medialive:DeleteEventBridgeRuleTemplate
	medialive:DeleteEventBridgeRuleTemplateGroup
	medialive:DeleteInput
	medialive:DeleteInputSecurityGroup
	medialive:DeleteMultiplex
	medialive:DeleteMultiplexProgram
	medialive:DeleteNetwork
	medialive:DeleteNode
	medialive:DeleteReservation
	medialive:DeleteSchedule
	medialive:DeleteSdiSource
	medialive:DeleteSignalMap
	medialive:DescribeAccountConfiguration

Prefixo do serviço	Ações do
	medialive:DescribeChannel
	medialive:DescribeChannelPlacementGroup
	medialive:DescribeCluster
	medialive:DescribeInput
	medialive:DescribeInputDevice
	medialive:DescribeInputDeviceThumbnail
	medialive:DescribeInputSecurityGroup
	medialive:DescribeMultiplex
	medialive:DescribeMultiplexProgram
	medialive:DescribeNetwork
	medialive:DescribeNode
	medialive:DescribeOffering
	medialive:DescribeReservation
	medialive:DescribeSchedule
	medialive:DescribeSdiSource
	medialive:DescribeThumbnails
	medialive:GetCloudWatchAlarmTemplate
	medialive:GetCloudWatchAlarmTemplateGroup
	medialive:GetEventBridgeRuleTemplate
	medialive:GetEventBridgeRuleTemplateGroup
	medialive:GetSignalMap

Prefixo do serviço	Ações do
	medialive:ListAlerts
	medialive:ListChannelPlacementGroups
	medialive:ListChannels
	medialive:ListCloudWatchAlarmTemplateGroups
	medialive:ListCloudWatchAlarmTemplates
	medialive:ListClusterAlerts
	medialive:ListClusters
	medialive:ListEventBridgeRuleTemplateGroups
	medialive:ListEventBridgeRuleTemplates
	medialive:ListInputDeviceTransfers
	medialive:ListInputDevices
	medialive:ListInputSecurityGroups
	medialive:ListInputs
	medialive:ListMultiplexAlerts
	medialive:ListMultiplexPrograms
	medialive:ListMultiplexes
	medialive:ListNetworks
	medialive:ListNodes
	medialive:ListOfferings
	medialive:ListReservations
	medialive:ListSdiSources

Prefixo do serviço	Ações do
	medialive:ListSignalMaps
	medialive:ListVersions
	medialive:PurchaseOffering
	medialive:RebootInputDevice
	medialive:RejectInputDeviceTransfer
	medialive:RestartChannelPipelines
	medialive:StartChannel
	medialive:StartDeleteMonitorDeployment
	medialive:StartInputDevice
	medialive:StartInputDeviceMaintenanceWindow
	medialive:StartMonitorDeployment
	medialive:StartMultiplex
	medialive:StartUpdateSignalMap
	medialive:StopChannel
	medialive:StopInputDevice
	medialive:StopMultiplex
	medialive:TransferInputDevice
	medialive:UpdateAccountConfiguration
	medialive:UpdateChannel
	medialive:UpdateChannelClass
	medialive:UpdateChannelPlacementGroup

Prefixo do serviço	Ações do
	medialive:UpdateCloudWatchAlarmTemplate
	medialive:UpdateCloudWatchAlarmTemplateGroup
	medialive:UpdateCluster
	medialive:UpdateEventBridgeRuleTemplate
	medialive:UpdateEventBridgeRuleTemplateGroup
	medialive:UpdateInput
	medialive:UpdateInputDevice
	medialive:UpdateInputSecurityGroup
	medialive:UpdateMultiplex
	medialive:UpdateMultiplexProgram
	medialive:UpdateNetwork
	medialive:UpdateNode
	medialive:UpdateNodeState
	medialive:UpdateReservation
	medialive:UpdateSdiSource

Prefixo do serviço	Ações do
mediatailor	mediatailor:ConfigureLogsForPlaybackConfiguration mediatailor:CreateChannel mediatailor:CreateLiveSource mediatailor:CreatePrefetchSchedule mediatailor:CreateProgram mediatailor:CreateSourceLocation mediatailor:CreateVodSource mediatailor:DeleteChannel mediatailor:DeleteChannelPolicy mediatailor:DeleteLiveSource mediatailor:DeletePlaybackConfiguration mediatailor:DeletePrefetchSchedule mediatailor:DeleteProgram mediatailor:DeleteSourceLocation mediatailor:DeleteVodSource mediatailor:DescribeChannel mediatailor:DescribeLiveSource mediatailor:DescribeProgram mediatailor:DescribeSourceLocation mediatailor:DescribeVodSource mediatailor:GetChannelPolicy

Prefixo do serviço	Ações do
	mediatailor:GetChannelSchedule mediatailor:GetPlaybackConfiguration mediatailor:GetPrefetchSchedule mediatailor:ListAlerts mediatailor:ListChannels mediatailor:ListLiveSources mediatailor:ListPlaybackConfigurations mediatailor:ListPrefetchSchedules mediatailor:ListSourceLocations mediatailor:ListVodSources mediatailor:PutChannelPolicy mediatailor:PutPlaybackConfiguration mediatailor:StartChannel mediatailor:StopChannel mediatailor:UpdateChannel mediatailor:UpdateLiveSource mediatailor:UpdateProgram mediatailor:UpdateSourceLocation mediatailor:UpdateVodSource

Prefixo do serviço	Ações do
memorydb	memorydb:BatchUpdateCluster
	memorydb:CopySnapshot
	memorydb:CreateAcl
	memorydb:CreateCluster
	memorydb:CreateMultiRegionCluster
	memorydb:CreateParameterGroup
	memorydb:CreateSnapshot
	memorydb:CreateSubnetGroup
	memorydb:CreateUser
	memorydb:DeleteAcl
	memorydb:DeleteCluster
	memorydb:DeleteMultiRegionCluster
	memorydb:DeleteParameterGroup
	memorydb:DeleteSnapshot
	memorydb:DeleteSubnetGroup
	memorydb:DeleteUser
	memorydb:DescribeAcls
	memorydb:DescribeClusters
	memorydb:DescribeEngineVersions
	memorydb:DescribeEvents
	memorydb:DescribeMultiRegionClusters

Prefixo do serviço	Ações do
	memorydb:DescribeMultiRegionParameterGroups
	memorydb:DescribeMultiRegionParameters
	memorydb:DescribeParameterGroups
	memorydb:DescribeParameters
	memorydb:DescribeReservedNodes
	memorydb:DescribeReservedNodesOfferings
	memorydb:DescribeServiceUpdates
	memorydb:DescribeSnapshots
	memorydb:DescribeSubnetGroups
	memorydb:DescribeUsers
	memorydb:FailoverShard
	memorydb:ListAllowedMultiRegionClusterUpdates
	memorydb:ListAllowedNodeTypeUpdates
	memorydb:PurchaseReservedNodesOffering
	memorydb:ResetParameterGroup
	memorydb:UpdateAcl
	memorydb:UpdateCluster
	memorydb:UpdateMultiRegionCluster
	memorydb:UpdateParameterGroup
	memorydb:UpdateSubnetGroup
	memorydb:UpdateUser

Prefixo do serviço	Ações do
mgh	mgh:AssociateCreatedArtifact mgh:AssociateDiscoveredResource mgh:AssociateSourceResource mgh:CreateHomeRegionControl mgh:CreateProgressUpdateStream mgh:DeleteHomeRegionControl mgh:DeleteProgressUpdateStream mgh:DescribeApplicationState mgh:DescribeHomeRegionControls mgh:DescribeMigrationTask mgh:DisassociateCreatedArtifact mgh:DisassociateDiscoveredResource mgh:DisassociateSourceResource mgh:GetHomeRegion mgh:ImportMigrationTask mgh:ListApplicationStates mgh:ListCreatedArtifacts mgh:ListDiscoveredResources mgh:ListMigrationTaskUpdates mgh:ListMigrationTasks mgh:ListProgressUpdateStreams

Prefixo do serviço	Ações do
	mgh:ListSourceResources mgh:NotifyApplicationState mgh:NotifyMigrationTaskState mgh:PutResourceAttributes

Prefixo do serviço	Ações do
mgn	mgn:ArchiveApplication
	mgn:ArchiveWave
	mgn:AssociateApplications
	mgn:AssociateSourceServers
	mgn:ChangeServerLifeCycleState
	mgn:CreateApplication
	mgn:CreateConnector
	mgn:CreateLaunchConfigurationTemplate
	mgn:CreateNetworkMigrationDefinition
	mgn:CreateReplicationConfigurationTemplate
	mgn:CreateWave
	mgn:DeleteApplication
	mgn:DeleteConnector
	mgn:DeleteJob
	mgn:DeleteLaunchConfigurationTemplate
	mgn:DeleteNetworkMigrationDefinition
	mgn:DeleteReplicationConfigurationTemplate
	mgn:DeleteSourceServer
	mgn:DeleteVcenterClient
	mgn:DeleteWave
	mgn:DescribeJobLogItems

Prefixo do serviço	Ações do
	mgn:DescribeJobs
	mgn:DescribeLaunchConfigurationTemplates
	mgn:DescribeReplicationConfigurationTemplates
	mgn:DescribeVcenterClients
	mgn:DisassociateApplications
	mgn:DisassociateSourceServers
	mgn:DisconnectFromService
	mgn:FinalizeCutover
	mgn:GetNetworkMigrationDefinition
	mgn:GetNetworkMigrationMapperSegmentConstruct
	mgn:GetReplicationConfiguration
	mgn:InitializeService
	mgn:ListConnectors
	mgn:ListExportErrors
	mgn:ListExports
	mgn:ListImportErrors
	mgn:ListImportFileEnrichments
	mgn:ListImports
	mgn:ListManagedAccounts
	mgn:ListNetworkMigrationAnalyses
	mgn:ListNetworkMigrationAnalysisResults

Prefixo do serviço	Ações do
	mgn:ListNetworkMigrationCodeGenerationSegments
	mgn:ListNetworkMigrationCodeGenerations
	mgn:ListNetworkMigrationDefinitions
	mgn:ListNetworkMigrationDeployedStacks
	mgn:ListNetworkMigrationDeployments
	mgn:ListNetworkMigrationExecutions
	mgn:ListNetworkMigrationMapperSegmentConstructs
	mgn:ListNetworkMigrationMapperSegments
	mgn:ListNetworkMigrationMappingUpdates
	mgn:ListNetworkMigrationMappings
	mgn:ListSourceServerActions
	mgn:ListTemplateActions
	mgn:MarkAsArchived
	mgn:PauseReplication
	mgn:PutSourceServerAction
	mgn:PutTemplateAction
	mgn:RemoveSourceServerAction
	mgn:RemoveTemplateAction
	mgn:ResumeReplication
	mgn:RetryDataReplication
	mgn:StartCutover

Prefixo do serviço	Ações do
	mgn:StartExport
	mgn:StartImport
	mgn:StartImportFileEnrichment
	mgn:StartNetworkMigrationCodeGeneration
	mgn:StartNetworkMigrationMapping
	mgn:StartNetworkMigrationMappingUpdate
	mgn:StartReplication
	mgn:StartTest
	mgn:StopReplication
	mgn:TerminateTargetInstances
	mgn:UnarchiveApplication
	mgn:UnarchiveWave
	mgn:UpdateApplication
	mgn:UpdateConnector
	mgn:UpdateLaunchConfigurationTemplate
	mgn:UpdateNetworkMigrationDefinition
	mgn:UpdateNetworkMigrationMapperSegment
	mgn:UpdateReplicationConfiguration
	mgn:UpdateReplicationConfigurationTemplate
	mgn:UpdateSourceServer
	mgn:UpdateSourceServerReplicationType

Prefixo do serviço	Ações do
	mgn:UpdateWave

Prefixo do serviço	Ações do
migrationhub-strategy	migrationhub-strategy:GetAntiPattern migrationhub-strategy:GetApplicationComponentDetails migrationhub-strategy:GetApplicationComponentStrategies migrationhub-strategy:GetAssessment migrationhub-strategy:GetImportFileTask migrationhub-strategy:GetLatestAssessmentId migrationhub-strategy:GetMessage migrationhub-strategy:GetPortfolioPreferences migrationhub-strategy:GetPortfolioSummary migrationhub-strategy:GetRecommendationReportDetails migrationhub-strategy:GetServerDetails migrationhub-strategy:GetServerStrategies migrationhub-strategy:ListAnalyzableServers migrationhub-strategy:ListAntiPatterns migrationhub-strategy:ListApplicationComponents migrationhub-strategy:ListCollectors migrationhub-strategy:ListImportFileTask migrationhub-strategy:ListJarArtifacts migrationhub-strategy:ListServers migrationhub-strategy:PutLogData migrationhub-strategy:PutMetricData

Prefixo do serviço	Ações do
	migrationhub-strategy:PutPortfolioPreferences migrationhub-strategy:RegisterCollector migrationhub-strategy:SendMessage migrationhub-strategy:StartAssessment migrationhub-strategy:StartImportFileTask migrationhub-strategy:StartRecommendationReportGeneration migrationhub-strategy:StopAssessment migrationhub-strategy:UpdateApplicationComponentConfig migrationhub-strategy:UpdateCollectorConfiguration migrationhub-strategy:UpdateServerConfig

Prefixo do serviço	Ações do
mobiletargeting	mobiletargeting:CreateApp
	mobiletargeting:CreateCampaign
	mobiletargeting:CreateEmailTemplate
	mobiletargeting:CreateExportJob
	mobiletargeting:CreateImportJob
	mobiletargeting:CreateInAppTemplate
	mobiletargeting:CreateJourney
	mobiletargeting:CreatePushTemplate
	mobiletargeting:CreateRecommenderConfiguration
	mobiletargeting:CreateSegment
	mobiletargeting:CreateSmsTemplate
	mobiletargeting:CreateVoiceTemplate
	mobiletargeting:DeleteAdmChannel
	mobiletargeting:DeleteApnsChannel
	mobiletargeting:DeleteApnsSandboxChannel
	mobiletargeting:DeleteApnsVoipChannel
	mobiletargeting:DeleteApnsVoipSandboxChannel
	mobiletargeting:DeleteApp
	mobiletargeting:DeleteBaiduChannel
	mobiletargeting:DeleteCampaign
	mobiletargeting:DeleteEmailChannel

Prefixo do serviço	Ações do
	mobiletargeting:DeleteEmailTemplate
	mobiletargeting:DeleteEndpoint
	mobiletargeting:DeleteEventStream
	mobiletargeting:DeleteGcmChannel
	mobiletargeting:DeleteInAppTemplate
	mobiletargeting:DeleteJourney
	mobiletargeting:DeletePushTemplate
	mobiletargeting:DeleteRecommenderConfiguration
	mobiletargeting:DeleteSegment
	mobiletargeting:DeleteSmsChannel
	mobiletargeting:DeleteSmsTemplate
	mobiletargeting:DeleteUserEndpoints
	mobiletargeting:DeleteVoiceChannel
	mobiletargeting:DeleteVoiceTemplate
	mobiletargeting:GetAdmChannel
	mobiletargeting:GetApnsChannel
	mobiletargeting:GetApnsSandboxChannel
	mobiletargeting:GetApnsVoipChannel
	mobiletargeting:GetApnsVoipSandboxChannel
	mobiletargeting:GetApp
	mobiletargeting:GetApplicationDateRangeKpi

Prefixo do serviço	Ações do
	mobiletargeting:GetApplicationSettings
	mobiletargeting:GetApps
	mobiletargeting:GetBaiduChannel
	mobiletargeting:GetCampaign
	mobiletargeting:GetCampaignActivities
	mobiletargeting:GetCampaignDateRangeKpi
	mobiletargeting:GetCampaignVersion
	mobiletargeting:GetCampaignVersions
	mobiletargeting:GetCampaigns
	mobiletargeting:GetChannels
	mobiletargeting:GetEmailChannel
	mobiletargeting:GetEmailTemplate
	mobiletargeting:GetEndpoint
	mobiletargeting:GetEventStream
	mobiletargeting:GetExportJob
	mobiletargeting:GetExportJobs
	mobiletargeting:GetGcmChannel
	mobiletargeting:GetImportJob
	mobiletargeting:GetImportJobs
	mobiletargeting:GetInAppMessages
	mobiletargeting:GetInAppTemplate

Prefixo do serviço	Ações do
	mobiletargeting:GetJourney
	mobiletargeting:GetJourneyDateRangeKpi
	mobiletargeting:GetJourneyExecutionActivityMetrics
	mobiletargeting:GetJourneyExecutionMetrics
	mobiletargeting:GetJourneyRunExecutionActivityMetrics
	mobiletargeting:GetJourneyRunExecutionMetrics
	mobiletargeting:GetJourneyRuns
	mobiletargeting:GetPushTemplate
	mobiletargeting:GetRecommenderConfiguration
	mobiletargeting:GetRecommenderConfigurations
	mobiletargeting:GetSegment
	mobiletargeting:GetSegmentExportJobs
	mobiletargeting:GetSegmentImportJobs
	mobiletargeting:GetSegmentVersion
	mobiletargeting:GetSegmentVersions
	mobiletargeting:GetSegments
	mobiletargeting:GetSmsChannel
	mobiletargeting:GetSmsTemplate
	mobiletargeting:GetUserEndpoints
	mobiletargeting:GetVoiceChannel
	mobiletargeting:GetVoiceTemplate

Prefixo do serviço	Ações do
	mobiletargeting:ListJourneys
	mobiletargeting:ListTemplateVersions
	mobiletargeting:ListTemplates
	mobiletargeting:PhoneNumberValidate
	mobiletargeting:PutEventStream
	mobiletargeting:RemoveAttributes
	mobiletargeting:UpdateAdmChannel
	mobiletargeting:UpdateApnsChannel
	mobiletargeting:UpdateApnsSandboxChannel
	mobiletargeting:UpdateApnsVoipChannel
	mobiletargeting:UpdateApnsVoipSandboxChannel
	mobiletargeting:UpdateApplicationSettings
	mobiletargeting:UpdateBaiduChannel
	mobiletargeting:UpdateCampaign
	mobiletargeting:UpdateEmailChannel
	mobiletargeting:UpdateEmailTemplate
	mobiletargeting:UpdateEndpoint
	mobiletargeting:UpdateEndpointsBatch
	mobiletargeting:UpdateGcmChannel
	mobiletargeting:UpdateInAppTemplate
	mobiletargeting:UpdateJourney

Prefixo do serviço	Ações do
	mobiletargeting:UpdateJourneyState
	mobiletargeting:UpdatePushTemplate
	mobiletargeting:UpdateRecommenderConfiguration
	mobiletargeting:UpdateSegment
	mobiletargeting:UpdateSmsChannel
	mobiletargeting:UpdateSmsTemplate
	mobiletargeting:UpdateTemplateActiveVersion
	mobiletargeting:UpdateVoiceChannel
	mobiletargeting:UpdateVoiceTemplate
	mobiletargeting:VerifyOTPMessage

Prefixo do serviço	Ações do
mq	mq:CreateBroker mq:CreateConfiguration mq:CreateUser mq:DeleteBroker mq:DeleteConfiguration mq:DeleteUser mq:DescribeBroker mq:DescribeBrokerEngineTypes mq:DescribeBrokerInstanceOptions mq:DescribeConfiguration mq:DescribeConfigurationRevision mq:DescribeSharedResources mq:DescribeUser mq:ListBrokers mq:ListConfigurationRevisions mq:ListConfigurations mq:ListUsers mq:Promote mq:RebootBroker mq:UpdateBroker mq:UpdateConfiguration

Prefixo do serviço	Ações do
	mq:UpdateUser

Prefixo do serviço	Ações do
networkmanager	networkmanager:AcceptAttachment
	networkmanager:AssociateConnectPeer
	networkmanager:AssociateCustomerGateway
	networkmanager:AssociateLink
	networkmanager:AssociateTransitGatewayConnectPeer
	networkmanager:CreateConnectAttachment
	networkmanager:CreateConnectPeer
	networkmanager:CreateConnection
	networkmanager:CreateCoreNetwork
	networkmanager:CreateDevice
	networkmanager:CreateDirectConnectGatewayAttachment
	networkmanager:CreateGlobalNetwork
	networkmanager:CreateLink
	networkmanager:CreateSite
	networkmanager:CreateSiteToSiteVpnAttachment
	networkmanager:CreateTransitGatewayPeering
	networkmanager:CreateTransitGatewayRouteTableAttachment
	networkmanager:CreateVpcAttachment
	networkmanager:DeleteAttachment
	networkmanager:DeleteConnectPeer
	networkmanager:DeleteConnection

Prefixo do serviço	Ações do
	networkmanager:DeleteCoreNetwork
	networkmanager:DeleteCoreNetworkPolicyVersion
	networkmanager:DeleteDevice
	networkmanager:DeleteGlobalNetwork
	networkmanager:DeleteLink
	networkmanager:DeletePeering
	networkmanager:DeleteResourcePolicy
	networkmanager:DeleteSite
	networkmanager:DeregisterTransitGateway
	networkmanager:DescribeGlobalNetworks
	networkmanager:DisassociateConnectPeer
	networkmanager:DisassociateCustomerGateway
	networkmanager:DisassociateLink
	networkmanager:DisassociateTransitGatewayConnectPeer
	networkmanager:ExecuteCoreNetworkChangeSet
	networkmanager:GetConnectAttachment
	networkmanager:GetConnectPeer
	networkmanager:GetConnectPeerAssociations
	networkmanager:GetConnections
	networkmanager:GetCoreNetwork
	networkmanager:GetCoreNetworkChangeEvents

Prefixo do serviço	Ações do
	networkmanager:GetCoreNetworkChangeSet
	networkmanager:GetCoreNetworkPolicy
	networkmanager:GetCustomerGatewayAssociations
	networkmanager:GetDevices
	networkmanager:GetLinkAssociations
	networkmanager:GetLinks
	networkmanager:GetNetworkResourceCounts
	networkmanager:GetNetworkResourceRelationships
	networkmanager:GetNetworkResources
	networkmanager:GetNetworkRoutes
	networkmanager:GetNetworkTelemetry
	networkmanager:GetResourcePolicy
	networkmanager:GetRouteAnalysis
	networkmanager:GetSiteToSiteVpnAttachment
	networkmanager:GetSites
	networkmanager:GetTransitGatewayConnectPeerAssociations
	networkmanager:GetTransitGatewayPeering
	networkmanager:GetTransitGatewayRegistrations
	networkmanager:GetTransitGatewayRouteTableAttachment
	networkmanager:GetVpcAttachment
	networkmanager:ListAttachmentRoutingPolicyAssociations

Prefixo do serviço	Ações do
	networkmanager:ListAttachments
	networkmanager:ListConnectPeers
	networkmanager:ListCoreNetworkPolicyVersions
	networkmanager:ListCoreNetworkPrefixListAssociations
	networkmanager:ListCoreNetworkRoutingInformation
	networkmanager:ListCoreNetworks
	networkmanager:ListOrganizationServiceAccessStatus
	networkmanager:ListPeerings
	networkmanager:PutAttachmentRoutingPolicyLabel
	networkmanager:PutCoreNetworkPolicy
	networkmanager:PutResourcePolicy
	networkmanager:RegisterTransitGateway
	networkmanager:RejectAttachment
	networkmanager:RemoveAttachmentRoutingPolicyLabel
	networkmanager:RestoreCoreNetworkPolicyVersion
	networkmanager:StartOrganizationServiceAccessUpdate
	networkmanager:StartRouteAnalysis
	networkmanager:UpdateConnection
	networkmanager:UpdateCoreNetwork
	networkmanager:UpdateDevice
	networkmanager:UpdateDirectConnectGatewayAttachment

Prefixo do serviço	Ações do
	networkmanager:UpdateGlobalNetwork networkmanager:UpdateLink networkmanager:UpdateNetworkResourceMetadata networkmanager:UpdateSite networkmanager:UpdateVpcAttachment

Prefixo do serviço	Ações do
nimble	nimble:AcceptEulas nimble:CreateLaunchProfile nimble:CreateStreamingImage nimble:CreateStreamingSession nimble:CreateStreamingSessionStream nimble:CreateStudio nimble:CreateStudioComponent nimble:DeleteLaunchProfile nimble:DeleteLaunchProfileMember nimble:DeleteStreamingImage nimble:DeleteStreamingSession nimble:DeleteStudio nimble:DeleteStudioComponent nimble:DeleteStudioMember nimble:GetEula nimble:GetLaunchProfileDetails nimble:GetStreamingImage nimble:GetStreamingSession nimble:GetStreamingSessionBackup nimble:GetStreamingSessionStream nimble:GetStudio

Prefixo do serviço	Ações do
	nimble:GetStudioComponent nimble:GetStudioMember nimble:ListEulas nimble:ListLaunchProfileMembers nimble:ListLaunchProfiles nimble:ListStreamingImages nimble:ListStreamingSessionBackups nimble:ListStreamingSessions nimble:ListStudioComponents nimble:ListStudioMembers nimble:ListStudios nimble:PutLaunchProfileMembers nimble:PutStudioMembers nimble:StartStreamingSession nimble:StartStudioSSOConfigurationRepair nimble:StopStreamingSession nimble:UpdateLaunchProfile nimble:UpdateLaunchProfileMember nimble:UpdateStreamingImage nimble:UpdateStudio nimble:UpdateStudioComponent

Prefixo do serviço	Ações do
omics	omics:AbortMultipartReadSetUpload
	omics:AcceptShare
	omics:BatchDeleteReadSet
	omics:CancelAnnotationImportJob
	omics:CancelRun
	omics:CancelRunBatch
	omics:CancelVariantImportJob
	omics:CompleteMultipartReadSetUpload
	omics:CreateAnnotationStore
	omics:CreateAnnotationStoreVersion
	omics:CreateConfiguration
	omics:CreateMultipartReadSetUpload
	omics:CreateReferenceStore
	omics:CreateRunGroup
	omics:CreateSequenceStore
	omics:CreateShare
	omics:CreateVariantStore
	omics:CreateWorkflow
	omics:CreateWorkflowVersion
	omics:DeleteAnnotationStore
	omics:DeleteAnnotationStoreVersions

Prefixo do serviço	Ações do
	omics:DeleteBatch
	omics:DeleteConfiguration
	omics:DeleteReference
	omics:DeleteReferenceStore
	omics:DeleteRun
	omics:DeleteRunBatch
	omics:DeleteRunGroup
	omics:DeleteSequenceStore
	omics:DeleteShare
	omics:DeleteVariantStore
	omics:DeleteWorkflow
	omics:DeleteWorkflowVersion
	omics:GetAnnotationImportJob
	omics:GetAnnotationStore
	omics:GetAnnotationStoreVersion
	omics:GetBatch
	omics:GetConfiguration
	omics:GetReadSet
	omics:GetReadSetActivationJob
	omics:GetReadSetExportJob
	omics:GetReadSetImportJob

Prefixo do serviço	Ações do
	omics:GetReadSetMetadata
	omics:GetReference
	omics:GetReferenceImportJob
	omics:GetReferenceMetadata
	omics:GetReferenceStore
	omics:GetRun
	omics:GetRunGroup
	omics:GetRunTask
	omics:GetSequenceStore
	omics:GetShare
	omics:GetVariantImportJob
	omics:GetVariantStore
	omics:GetWorkflow
	omics:GetWorkflowVersion
	omics:ListAnnotationImportJobs
	omics:ListAnnotationStoreVersions
	omics:ListAnnotationStores
	omics:ListBatch
	omics:ListConfigurations
	omics:ListMultipartReadSetUploads
	omics:ListReadSetActivationJobs

Prefixo do serviço	Ações do
	omics:ListReadSetExportJobs
	omics:ListReadSetImportJobs
	omics:ListReadSetUploadParts
	omics:ListReadSets
	omics:ListReferenceImportJobs
	omics:ListReferenceStores
	omics:ListReferences
	omics:ListRunGroups
	omics:ListRunTasks
	omics:ListRuns
	omics:ListRunsInBatch
	omics:ListSequenceStores
	omics:ListShares
	omics:ListVariantImportJobs
	omics:ListVariantStores
	omics:ListWorkflowVersions
	omics:ListWorkflows
	omics:StartAnnotationImportJob
	omics:StartReadSetActivationJob
	omics:StartReadSetExportJob
	omics:StartReadSetImportJob

Prefixo do serviço	Ações do
	omics:StartReferenceImportJob
	omics:StartRun
	omics:StartRunBatch
	omics:StartVariantImportJob
	omics:UpdateAnnotationStore
	omics:UpdateAnnotationStoreVersion
	omics:UpdateRunGroup
	omics:UpdateVariantStore
	omics:UpdateWorkflow
	omics:UpdateWorkflowVersion
	omics:UploadReadSetPart

Prefixo do serviço	Ações do
opsworks	opsworks:AssignInstance opsworks:AssignVolume opsworks:AssociateElasticIp opsworks:AttachElasticLoadBalancer opsworks:CloneStack opsworks:CreateApp opsworks:CreateDeployment opsworks:CreateInstance opsworks:CreateLayer opsworks:CreateStack opsworks:CreateUserProfile opsworks:DeleteApp opsworks:DeleteInstance opsworks:DeleteLayer opsworks:DeleteStack opsworks:DeleteUserProfile opsworks:DeregisterEcsCluster opsworks:DeregisterElasticIp opsworks:DeregisterInstance opsworks:DeregisterRdsDbInstance opsworks:DeregisterVolume

Prefixo do serviço	Ações do
	opsworks:DescribeAgentVersions opsworks:DescribeApps opsworks:DescribeCommands opsworks:DescribeDeployments opsworks:DescribeEcsClusters opsworks:DescribeElasticIps opsworks:DescribeElasticLoadBalancers opsworks:DescribeInstances opsworks:DescribeLayers opsworks:DescribeLoadBasedAutoScaling opsworks:DescribeMyUserProfile opsworks:DescribeOperatingSystems opsworks:DescribePermissions opsworks:DescribeRaidArrays opsworks:DescribeRdsDbInstances opsworks:DescribeServiceErrors opsworks:DescribeStackProvisioningParameters opsworks:DescribeStackSummary opsworks:DescribeStacks opsworks:DescribeTimeBasedAutoScaling opsworks:DescribeUserProfiles

Prefixo do serviço	Ações do
	opsworks:DescribeVolumes opsworks:DetachElasticLoadBalancer opsworks:DisassociateElasticIp opsworks:GetHostnameSuggestion opsworks:GrantAccess opsworks:RebootInstance opsworks:RegisterEcsCluster opsworks:RegisterElasticIp opsworks:RegisterInstance opsworks:RegisterRdsDbInstance opsworks:RegisterVolume opsworks:SetLoadBasedAutoScaling opsworks:SetPermission opsworks:SetTimeBasedAutoScaling opsworks:StartInstance opsworks:StartStack opsworks:StopInstance opsworks:StopStack opsworks:UnassignInstance opsworks:UnassignVolume opsworks:UpdateApp

Prefixo do serviço	Ações do
	opsworks:UpdateElasticIp opsworks:UpdateInstance opsworks:UpdateLayer opsworks:UpdateMyUserProfile opsworks:UpdateRdsDbInstance opsworks:UpdateStack opsworks:UpdateUserProfile opsworks:UpdateVolume

Prefixo do serviço	Ações do
opsworks-cm	opsworks-cm:AssociateNode
	opsworks-cm:CreateBackup
	opsworks-cm:CreateServer
	opsworks-cm:DeleteBackup
	opsworks-cm:DeleteServer
	opsworks-cm:DescribeAccountAttributes
	opsworks-cm:DescribeBackups
	opsworks-cm:DescribeEvents
	opsworks-cm:DescribeNodeAssociationStatus
	opsworks-cm:DescribeServers
	opsworks-cm:DisassociateNode
	opsworks-cm:ExportServerEngineAttribute
	opsworks-cm:RestoreServer
	opsworks-cm:StartMaintenance
	opsworks-cm:UpdateServer
	opsworks-cm:UpdateServerEngineAttributes

Prefixo do serviço	Ações do
organizações	organizations:AcceptHandshake organizations:AttachPolicy organizations:CancelHandshake organizations:CloseAccount organizations:CreateAccount organizations:CreateGovCloudAccount organizations:CreateOrganization organizations:CreateOrganizationalUnit organizations:CreatePolicy organizations:DeclineHandshake organizations:DeleteOrganization organizations:DeleteOrganizationalUnit organizations:DeletePolicy organizations:DeleteResourcePolicy organizations:DeregisterDelegatedAdministrator organizations:DescribeAccount organizations:DescribeCreateAccountStatus organizations:DescribeEffectivePolicy organizations:DescribeHandshake organizations:DescribeOrganization organizations:DescribeOrganizationalUnit

Prefixo do serviço	Ações do
	organizations:DescribePolicy organizations:DescribeResourcePolicy organizations:DescribeResponsibilityTransfer organizations:DetachPolicy organizations:DisableAWSServiceAccess organizations:DisablePolicyType organizations:EnableAWSServiceAccess organizations:EnableAllFeatures organizations:EnablePolicyType organizations:InviteAccountToOrganization organizations:LeaveOrganization organizations:ListAWSServiceAccessForOrganization organizations:ListAccounts organizations:ListAccountsForParent organizations:ListAccountsWithInvalidEffectivePolicy organizations:ListChildren organizations:ListCreateAccountStatus organizations:ListDelegatedAdministrators organizations:ListDelegatedServicesForAccount organizations:ListHandshakesForAccount organizations:ListHandshakesForOrganization

Prefixo do serviço	Ações do
	organizations:ListInboundResponsibilityTransfers organizations:ListOrganizationalUnitsForParent organizations:ListOutboundResponsibilityTransfers organizations:ListParents organizations:ListPolicies organizations:ListPoliciesForTarget organizations:ListRoots organizations:ListTargetsForPolicy organizations:MoveAccount organizations:PutResourcePolicy organizations:RegisterDelegatedAdministrator organizations:RemoveAccountFromOrganization organizations:TerminateResponsibilityTransfer organizations:UpdateOrganizationalUnit organizations:UpdatePolicy organizations:UpdateResponsibilityTransfer

Prefixo do serviço	Ações do
outposts	outposts:CancelCapacityTask
	outposts:CancelOrder
	outposts:CreateOrder
	outposts:CreateOutpost
	outposts:CreatePrivateConnectivityConfig
	outposts:CreateQuote
	outposts:CreateRenewal
	outposts:CreateSite
	outposts:DeleteOutpost
	outposts:DeleteQuote
	outposts:DeleteSite
	outposts:GetCapacityTask
	outposts:GetCatalogItem
	outposts:GetConnection
	outposts:GetOrder
	outposts:GetOutpost
	outposts:GetOutpostBillingInformation
	outposts:GetOutpostInstanceTypes
	outposts:GetOutpostSupportedInstanceTypes
	outposts:GetPrivateConnectivityConfig
	outposts:GetQuote

Prefixo do serviço	Ações do
	outposts:GetRenewalPricing
	outposts:GetSite
	outposts:GetSiteAddress
	outposts:ListAssetInstances
	outposts:ListAssets
	outposts:ListBlockingInstancesForCapacityTask
	outposts:ListCapacityTasks
	outposts:ListCatalogItems
	outposts:ListOrderableInstanceTypes
	outposts:ListOrders
	outposts:ListOutposts
	outposts:ListQuotes
	outposts:ListSites
	outposts:StartCapacityTask
	outposts:StartConnection
	outposts:UpdateOutpost
	outposts:UpdateQuote
	outposts:UpdateSite
	outposts:UpdateSiteAddress
	outposts:UpdateSiteRackPhysicalProperties

Prefixo do serviço	Ações do
pi	pi:CreatePerformanceAnalysisReport pi:DeletePerformanceAnalysisReport pi:DescribeDimensionKeys pi:GetDimensionKeyDetails pi:GetPerformanceAnalysisReport pi:GetResourceMetadata pi:GetResourceMetrics pi:ListAvailableResourceDimensions pi:ListAvailableResourceMetrics pi:ListPerformanceAnalysisReports
pipes	pipes:CreatePipe pipes:DeletePipe pipes:DescribePipe pipes:ListPipes pipes:StartPipe pipes:StopPipe pipes:UpdatePipe

Prefixo do serviço	Ações do
polly	polly:DeleteLexicon polly:DescribeVoices polly:GetLexicon polly:GetSpeechSynthesisTask polly:ListLexicons polly:ListSpeechSynthesisTasks polly:PutLexicon polly:StartSpeechSynthesisTask polly:SynthesizeSpeech

Prefixo do serviço	Ações do
perfil	profile:AddProfileKey
	profile:BatchGetCalculatedAttributeForProfile
	profile:BatchGetProfile
	profile:CreateCalculatedAttributeDefinition
	profile:CreateDomain
	profile:CreateEventStream
	profile:CreateProfile
	profile:CreateRecommender
	profile:CreateRecommenderFilter
	profile:CreateRecommenderSchema
	profile:CreateSegmentDefinition
	profile:CreateSegmentEstimate
	profile:CreateSegmentSnapshot
	profile:CreateUploadJob
	profile:DeleteCalculatedAttributeDefinition
	profile:DeleteDomain
	profile:DeleteDomainObjectType
	profile:DeleteEventStream
	profile:DeleteIntegration
	profile:DeleteProfile
	profile:DeleteProfileKey

Prefixo do serviço	Ações do
	profile:DeleteProfileObject
	profile:DeleteProfileObjectType
	profile:DeleteRecommender
	profile:DeleteRecommenderFilter
	profile:DeleteRecommenderSchema
	profile:DeleteSegmentDefinition
	profile:DeleteWorkflow
	profile:DetectProfileObjectType
	profile:GetAutoMergingPreview
	profile:GetCalculatedAttributeDefinition
	profile:GetCalculatedAttributeForProfile
	profile:GetDomain
	profile:GetDomainObjectType
	profile:GetEventStream
	profile:GetIdentityResolutionJob
	profile:GetIntegration
	profile:GetMatches
	profile:GetObjectTypeAttributeStatistics
	profile:GetProfileObjectType
	profile:GetProfileObjectTypeTemplate
	profile:GetProfileRecommendations

Prefixo do serviço	Ações do
	profile:GetRecommender
	profile:GetRecommenderFilter
	profile:GetRecommenderSchema
	profile:GetSegmentDefinition
	profile:GetSegmentEstimate
	profile:GetSegmentMembership
	profile:GetSegmentSnapshot
	profile:GetSimilarProfiles
	profile:GetUploadJob
	profile:GetUploadJobPath
	profile:GetWorkflow
	profile:GetWorkflowSteps
	profile:ListAccountIntegrations
	profile:ListCalculatedAttributeDefinitions
	profile:ListCalculatedAttributesForProfile
	profile:ListDomainLayouts
	profile:ListDomainObjectTypes
	profile:ListDomains
	profile:ListEventStreams
	profile:ListIdentityResolutionJobs
	profile:ListIntegrations

Prefixo do serviço	Ações do
	profile:ListObjectTypeAttributeValues
	profile:ListObjectTypeAttributes
	profile:ListProfileAttributeValues
	profile:ListProfileObjectTypeTemplates
	profile:ListProfileObjectTypes
	profile:ListProfileObjects
	profile:ListRecommenderFilters
	profile:ListRecommenderRecipes
	profile:ListRecommenderSchemas
	profile:ListRecommenders
	profile:ListRuleBasedMatches
	profile:ListSegmentDefinitions
	profile:ListUploadJobs
	profile:ListWorkflows
	profile:MergeProfiles
	profile:PutDomainObjectType
	profile:PutIntegration
	profile:PutProfileObject
	profile:PutProfileObjectType
	profile:SearchProfiles
	profile:StartRecommender

Prefixo do serviço	Ações do
	profile:StartUploadJob profile:StopRecommender profile:StopUploadJob profile:UpdateCalculatedAttributeDefinition profile:UpdateDomain profile:UpdateProfile profile:UpdateRecommender

Prefixo do serviço	Ações do
qldb	qldb:CancelJournalKinesisStream qldb:CreateLedger qldb:DeleteLedger qldb:DescribeJournalKinesisStream qldb:DescribeJournalS3Export qldb:DescribeLedger qldb:ExportJournalToS3 qldb:GetBlock qldb:GetDigest qldb:GetRevision qldb:ListJournalKinesisStreamsForLedger qldb:ListJournalS3Exports qldb:ListJournalS3ExportsForLedger qldb:ListLedgers qldb:StreamJournalToKinesis qldb:UpdateLedger qldb:UpdateLedgerPermissionsMode

Prefixo do serviço	Ações do
ram	ram:AcceptResourceShareInvitation
	ram:AssociateResourceShare
	ram:AssociateResourceSharePermission
	ram:CreatePermission
	ram:CreatePermissionVersion
	ram:CreateResourceShare
	ram:DeletePermission
	ram:DeletePermissionVersion
	ram:DeleteResourceShare
	ram:DisassociateResourceShare
	ram:DisassociateResourceSharePermission
	ram:EnableSharingWithAwsOrganization
	ram:GetPermission
	ram:GetResourcePolicies
	ram:GetResourceShareAssociations
	ram:GetResourceShareInvitations
	ram:GetResourceShares
	ram:ListPendingInvitationResources
	ram:ListPermissionAssociations
	ram:ListPermissionVersions
	ram:ListPermissions

Prefixo do serviço	Ações do
	ram:ListPrincipals ram:ListReplacePermissionAssociationsWork ram:ListResourceSharePermissions ram:ListResourceTypes ram:ListResources ram:ListSourceAssociations ram:PromotePermissionCreatedFromPolicy ram:PromoteResourceShareCreatedFromPolicy ram:RejectResourceShareInvitation ram:ReplacePermissionAssociations ram:SetDefaultPermissionVersion ram:UpdateResourceShare
rbin	rbin:CreateRule rbin:DeleteRule rbin:GetRule rbin:ListRules rbin:LockRule rbin:UnlockRule rbin:UpdateRule

Prefixo do serviço	Ações do
rds	rds:AddRoleToDBCluster
	rds:AddRoleToDBInstance
	rds:AddSourceIdentifierToSubscription
	rds:ApplyPendingMaintenanceAction
	rds:AuthorizeDBSecurityGroupIngress
	rds:BacktrackDBCluster
	rds:CancelExportTask
	rds:CopyDBClusterParameterGroup
	rds:CopyDBClusterSnapshot
	rds:CopyDBParameterGroup
	rds:CopyDBSnapshot
	rds:CopyOptionGroup
	rds:CreateCustomDBEngineVersion
	rds:CreateDBClusterParameterGroup
	rds:CreateDBParameterGroup
	rds:CreateDBProxy
	rds:CreateDBProxyEndpoint
	rds:CreateDBSecurityGroup
	rds:CreateDBSubnetGroup
	rds:CreateEventSubscription
	rds:CreateGlobalCluster

Prefixo do serviço	Ações do
	rds:CreateOptionGroup
	rds:DeleteBlueGreenDeployment
	rds:DeleteDBClusterAutomatedBackup
	rds:DeleteDBClusterParameterGroup
	rds:DeleteDBClusterSnapshot
	rds:DeleteDBInstanceAutomatedBackup
	rds:DeleteDBParameterGroup
	rds:DeleteDBProxy
	rds:DeleteDBProxyEndpoint
	rds:DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rds:DeleteDBSubnetGroup
	rds:DeleteEventSubscription
	rds:DeleteGlobalCluster
	rds:DeleteOptionGroup
	rds:DeregisterDBProxyTargets
	rds:DescribeAccountAttributes
	rds:DescribeBlueGreenDeployments
	rds:DescribeCertificates
	rds:DescribeDBClusterAutomatedBackups
	rds:DescribeDBClusterBacktracks

Prefixo do serviço	Ações do
	rds:DescribeDBClusterEndpoints
	rds:DescribeDBClusterParameterGroups
	rds:DescribeDBClusterParameters
	rds:DescribeDBClusterSnapshotAttributes
	rds:DescribeDBClusterSnapshots
	rds:DescribeDBClusters
	rds:DescribeDBEngineVersions
	rds:DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rds:DescribeDBLogFiles
	rds:DescribeDBMajorEngineVersions
	rds:DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds:DescribeDBProxyEndpoints
	rds:DescribeDBProxyTargetGroups
	rds:DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rds:DescribeDBSecurityGroups
	rds:DescribeDBSnapshotAttributes
	rds:DescribeDBSnapshotTenantDatabases

Prefixo do serviço	Ações do
	rds:DescribeDBSnapshots
	rds:DescribeDBSubnetGroups
	rds:DescribeEngineDefaultClusterParameters
	rds:DescribeEngineDefaultParameters
	rds:DescribeEventCategories
	rds:DescribeEventSubscriptions
	rds:DescribeEvents
	rds:DescribeExportTasks
	rds:DescribeGlobalClusters
	rds:DescribeIntegrations
	rds:DescribeOptionGroupOptions
	rds:DescribeOptionGroups
	rds:DescribeOrderableDBInstanceOptions
	rds:DescribePendingMaintenanceActions
	rds:DescribeReservedDBInstances
	rds:DescribeReservedDBInstancesOfferings
	rds:DescribeSourceRegions
	rds:DescribeTenantDatabases
	rds:DescribeValidDBInstanceModifications
	rds:DownloadCompleteDBLogFile
	rds:DownloadDBLogFilePortion

Prefixo do serviço	Ações do
	rds:FailoverDBCluster
	rds:FailoverGlobalCluster
	rds:ModifyActivityStream
	rds:ModifyCertificates
	rds:ModifyCurrentDBClusterCapacity
	rds:ModifyDBClusterEndpoint
	rds:ModifyDBClusterParameterGroup
	rds:ModifyDBClusterSnapshotAttribute
	rds:ModifyDBParameterGroup
	rds:ModifyDBProxy
	rds:ModifyDBProxyEndpoint
	rds:ModifyDBProxyTargetGroup
	rds:ModifyDBRecommendation
	rds:ModifyDBSnapshot
	rds:ModifyDBSnapshotAttribute
	rds:ModifyDBSubnetGroup
	rds:ModifyEventSubscription
	rds:ModifyGlobalCluster
	rds:ModifyOptionGroup
	rds:ModifyTenantDatabase
	rds:PurchaseReservedDBInstancesOffering

Prefixo do serviço	Ações do
	rds:RebootDBCluster
	rds:RegisterDBProxyTargets
	rds:RemoveFromGlobalCluster
	rds:RemoveRoleFromDBCluster
	rds:RemoveRoleFromDBInstance
	rds:RemoveSourceIdentifierFromSubscription
	rds:ResetDBClusterParameterGroup
	rds:ResetDBParameterGroup
	rds:RestoreDBClusterFromS3
	rds:RestoreDBClusterFromSnapshot
	rds:RestoreDBClusterToPointInTime
	rds:RestoreDBInstanceFromDBSnapshot
	rds:RestoreDBInstanceFromS3
	rds:RestoreDBInstanceToPointInTime
	rds:RevokeDBSecurityGroupIngress
	rds:StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rds:StartDBInstanceAutomatedBackupsReplication
	rds:StartExportTask
	rds:StopActivityStream

Prefixo do serviço	Ações do
	rds:StopDBCluster rds:StopDBInstance rds:StopDBInstanceAutomatedBackupsReplication rds:SwitchoverBlueGreenDeployment rds:SwitchoverGlobalCluster rds:SwitchoverReadReplica

Prefixo do serviço	Ações do
redshift	redshift:AcceptReservedNodeExchange redshift:AddPartner redshift:AssociateDataShareConsumer redshift:AuthorizeClusterSecurityGroupIngress redshift:AuthorizeDataShare redshift:AuthorizeEndpointAccess redshift:AuthorizeSnapshotAccess redshift:BatchDeleteClusterSnapshots redshift:BatchModifyClusterSnapshots redshift:CancelResize redshift:CopyClusterSnapshot redshift:CreateAuthenticationProfile redshift:CreateCluster redshift:CreateClusterParameterGroup redshift:CreateClusterSecurityGroup redshift:CreateClusterSnapshot redshift:CreateClusterSubnetGroup redshift:CreateCustomDomainAssociation redshift:CreateEndpointAccess redshift:CreateEventSubscription redshift:CreateHsmClientCertificate

Prefixo do serviço	Ações do
	redshift:CreateHsmConfiguration redshift:CreateIntegration redshift:CreateRedshiftIdcApplication redshift:CreateScheduledAction redshift:CreateSnapshotCopyGrant redshift:CreateSnapshotSchedule redshift:CreateUsageLimit redshift:DeauthorizeDataShare redshift:DeleteAuthenticationProfile redshift:DeleteCluster redshift:DeleteClusterParameterGroup redshift:DeleteClusterSecurityGroup redshift:DeleteClusterSnapshot redshift:DeleteClusterSubnetGroup redshift:DeleteCustomDomainAssociation redshift:DeleteEndpointAccess redshift:DeleteEventSubscription redshift:DeleteHsmClientCertificate redshift:DeleteHsmConfiguration redshift:DeletePartner redshift:DeleteRedshiftIdcApplication

Prefixo do serviço	Ações do
	redshift:DeleteResourcePolicy redshift:DeleteScheduledAction redshift:DeleteSnapshotCopyGrant redshift:DeleteSnapshotSchedule redshift:DeleteUsageLimit redshift:DeregisterNamespace redshift:DescribeAccountAttributes redshift:DescribeAuthenticationProfiles redshift:DescribeClusterDbRevisions redshift:DescribeClusterParameterGroups redshift:DescribeClusterParameters redshift:DescribeClusterSecurityGroups redshift:DescribeClusterSnapshots redshift:DescribeClusterSubnetGroups redshift:DescribeClusterTracks redshift:DescribeClusterVersions redshift:DescribeClusters redshift:DescribeCustomDomainAssociations redshift:DescribeDataShares redshift:DescribeDataSharesForConsumer redshift:DescribeDataSharesForProducer

Prefixo do serviço	Ações do
	redshift:DescribeDefaultClusterParameters redshift:DescribeEndpointAccess redshift:DescribeEndpointAuthorization redshift:DescribeEventCategories redshift:DescribeEventSubscriptions redshift:DescribeEvents redshift:DescribeHsmClientCertificates redshift:DescribeHsmConfigurations redshift:DescribeInboundIntegrations redshift:DescribeIntegrations redshift:DescribeLoggingStatus redshift:DescribeNodeConfigurationOptions redshift:DescribeOrderableClusterOptions redshift:DescribePartners redshift:DescribeRedshiftIdcApplications redshift:DescribeReservedNodeExchangeStatus redshift:DescribeReservedNodeOfferings redshift:DescribeReservedNodes redshift:DescribeResize redshift:DescribeScheduledActions redshift:DescribeSnapshotCopyGrants

Prefixo do serviço	Ações do
	redshift:DescribeSnapshotSchedules redshift:DescribeStorage redshift:DescribeTableRestoreStatus redshift:DescribeUsageLimits redshift:DisableLogging redshift:DisableSnapshotCopy redshift:DisassociateDataShareConsumer redshift:EnableLogging redshift:EnableSnapshotCopy redshift:FailoverPrimaryCompute redshift:GetClusterCredentials redshift:GetClusterCredentialsWithIAM redshift:GetIdentityCenterAuthToken redshift:GetReservedNodeExchangeConfigurationOptions redshift:GetReservedNodeExchangeOfferings redshift:GetResourcePolicy redshift:ListRecommendations redshift:ModifyAquaConfiguration redshift:ModifyAuthenticationProfile redshift:ModifyCluster redshift:ModifyClusterDbRevision

Prefixo do serviço	Ações do
	redshift:ModifyClusterIamRoles redshift:ModifyClusterMaintenance redshift:ModifyClusterParameterGroup redshift:ModifyClusterSnapshot redshift:ModifyClusterSnapshotSchedule redshift:ModifyClusterSubnetGroup redshift:ModifyCustomDomainAssociation redshift:ModifyEndpointAccess redshift:ModifyEventSubscription redshift:ModifyRedshiftIdcApplication redshift:ModifyScheduledAction redshift:ModifySnapshotCopyRetentionPeriod redshift:ModifySnapshotSchedule redshift:ModifyUsageLimit redshift:PauseCluster redshift:PurchaseReservedNodeOffering redshift:PutResourcePolicy redshift:RebootCluster redshift:RegisterNamespace redshift:RejectDataShare redshift:ResetClusterParameterGroup

Prefixo do serviço	Ações do
	redshift:ResizeCluster redshift:RestoreFromClusterSnapshot redshift:RestoreTableFromClusterSnapshot redshift:ResumeCluster redshift:RevokeClusterSecurityGroupIngress redshift:RevokeEndpointAccess redshift:RevokeSnapshotAccess redshift:RotateEncryptionKey redshift:UpdatePartnerStatus
redshift-data	redshift-data:BatchExecuteStatement redshift-data:CancelStatement redshift-data:DescribeStatement redshift-data:DescribeTable redshift-data:ExecuteStatement redshift-data:GetStatementResult redshift-data:ListDatabases redshift-data:ListSchemas redshift-data:ListSessions redshift-data:ListStatements redshift-data:ListTables

Prefixo do serviço	Ações do
refactor-spaces	refactor-spaces:CreateApplication refactor-spaces:CreateEnvironment refactor-spaces:CreateRoute refactor-spaces:CreateService refactor-spaces:DeleteApplication refactor-spaces:DeleteEnvironment refactor-spaces:DeleteResourcePolicy refactor-spaces:DeleteRoute refactor-spaces:DeleteService refactor-spaces:GetApplication refactor-spaces:GetEnvironment refactor-spaces:GetResourcePolicy refactor-spaces:GetRoute refactor-spaces:GetService refactor-spaces:ListApplications refactor-spaces:ListEnvironmentVpcs refactor-spaces:ListEnvironments refactor-spaces:ListRoutes refactor-spaces:ListServices refactor-spaces:PutResourcePolicy refactor-spaces:UpdateRoute

Prefixo do serviço	Ações do
rekognition	rekognition:AssociateFaces
	rekognition:CompareFaces
	rekognition:CopyProjectVersion
	rekognition:CreateCollection
	rekognition:CreateDataset
	rekognition:CreateFaceLivenessSession
	rekognition:CreateProject
	rekognition:CreateProjectVersion
	rekognition:CreateStreamProcessor
	rekognition:CreateUser
	rekognition:DeleteCollection
	rekognition:DeleteDataset
	rekognition:DeleteFaces
	rekognition:DeleteProject
	rekognition:DeleteProjectPolicy
	rekognition:DeleteProjectVersion
	rekognition:DeleteStreamProcessor
	rekognition:DeleteUser
	rekognition:DescribeCollection
	rekognition:DescribeDataset
	rekognition:DescribeProjectVersions

Prefixo do serviço	Ações do
	rekognition:DescribeProjects
	rekognition:DescribeStreamProcessor
	rekognition:DetectCustomLabels
	rekognition:DetectFaces
	rekognition:DetectLabels
	rekognition:DetectModerationLabels
	rekognition:DetectProtectiveEquipment
	rekognition:DetectText
	rekognition:DisassociateFaces
	rekognition:DistributeDatasetEntries
	rekognition:GetCelebrityInfo
	rekognition:GetCelebrityRecognition
	rekognition:GetContentModeration
	rekognition:GetFaceDetection
	rekognition:GetFaceLivenessSessionResults
	rekognition:GetFaceSearch
	rekognition:GetLabelDetection
	rekognition:GetMediaAnalysisJob
	rekognition:GetPersonTracking
	rekognition:GetSegmentDetection
	rekognition:GetTextDetection

Prefixo do serviço	Ações do
	rekognition:IndexFaces
	rekognition:ListCollections
	rekognition:ListDatasetEntries
	rekognition:ListDatasetLabels
	rekognition:ListFaces
	rekognition:ListMediaAnalysisJobs
	rekognition:ListProjectPolicies
	rekognition:ListStreamProcessors
	rekognition:ListUsers
	rekognition:PutProjectPolicy
	rekognition:RecognizeCelebrities
	rekognition:SearchFaces
	rekognition:SearchFacesByImage
	rekognition:SearchUsers
	rekognition:SearchUsersByImage
	rekognition:StartCelebrityRecognition
	rekognition:StartContentModeration
	rekognition:StartFaceDetection
	rekognition:StartFaceLivenessSession
	rekognition:StartFaceSearch
	rekognition:StartLabelDetection

Prefixo do serviço	Ações do
	rekognition:StartMediaAnalysisJob
	rekognition:StartPersonTracking
	rekognition:StartProjectVersion
	rekognition:StartSegmentDetection
	rekognition:StartStreamProcessor
	rekognition:StartTextDetection
	rekognition:StopProjectVersion
	rekognition:StopStreamProcessor
	rekognition:UpdateDatasetEntries
	rekognition:UpdateStreamProcessor

Prefixo do serviço	Ações do
resiliencehub	resiliencehub:AcceptResourceGroupingRecommendations resiliencehub:AddDraftAppVersionResourceMappings resiliencehub:BatchUpdateRecommendationStatus resiliencehub:CreateApp resiliencehub:CreateAppVersionAppComponent resiliencehub:CreateAppVersionResource resiliencehub:CreateRecommendationTemplate resiliencehub:CreateResiliencyPolicy resiliencehub:DeleteApp resiliencehub:DeleteAppAssessment resiliencehub:DeleteAppInputSource resiliencehub:DeleteAppVersionAppComponent resiliencehub:DeleteAppVersionResource resiliencehub:DeleteRecommendationTemplate resiliencehub:DeleteResiliencyPolicy resiliencehub:DescribeApp resiliencehub:DescribeAppAssessment resiliencehub:DescribeAppVersion resiliencehub:DescribeAppVersionAppComponent resiliencehub:DescribeAppVersionResource resiliencehub:DescribeAppVersionResourcesResolutionStatus

Prefixo do serviço	Ações do
	resiliencehub:DescribeAppVersionTemplate resiliencehub:DescribeDraftAppVersionResourcesImportStatus resiliencehub:DescribeMetricsExport resiliencehub:DescribeResiliencyPolicy resiliencehub:DescribeResourceGroupingRecommendationTask resiliencehub:ImportResourcesToDraftAppVersion resiliencehub:ListAlarmRecommendations resiliencehub:ListAppAssessmentComplianceDrifts resiliencehub:ListAppAssessmentResourceDrifts resiliencehub:ListAppAssessments resiliencehub:ListAppComponentCompliances resiliencehub:ListAppComponentRecommendations resiliencehub:ListAppInputSources resiliencehub:ListAppVersionAppComponents resiliencehub:ListAppVersionResourceMappings resiliencehub:ListAppVersionResources resiliencehub:ListAppVersions resiliencehub:ListApps resiliencehub:ListMetrics resiliencehub:ListRecommendationTemplates resiliencehub:ListResiliencyPolicies

Prefixo do serviço	Ações do
	resiliencehub:ListResourceGroupingRecommendations resiliencehub:ListSopRecommendations resiliencehub:ListSuggestedResiliencyPolicies resiliencehub:ListTestRecommendations resiliencehub:ListUnsupportedAppVersionResources resiliencehub:PublishAppVersion resiliencehub:PutDraftAppVersionTemplate resiliencehub:RejectResourceGroupingRecommendations resiliencehub:RemoveDraftAppVersionResourceMappings resiliencehub:ResolveAppVersionResources resiliencehub:StartAppAssessment resiliencehub:StartResourceGroupingRecommendationTask resiliencehub:UpdateApp resiliencehub:UpdateAppVersion resiliencehub:UpdateAppVersionAppComponent resiliencehub:UpdateAppVersionResource resiliencehub:UpdateResiliencyPolicy

Prefixo do serviço	Ações do
resource-explorer-2	resource-explorer-2:AssociateDefaultView
	resource-explorer-2:BatchGetView
	resource-explorer-2:CreateIndex
	resource-explorer-2:CreateResourceExplorerSetup
	resource-explorer-2:CreateView
	resource-explorer-2:DeleteIndex
	resource-explorer-2:DeleteResourceExplorerSetup
	resource-explorer-2:DeleteView
	resource-explorer-2:DisassociateDefaultView
	resource-explorer-2:GetAccountLevelServiceConfiguration
	resource-explorer-2:GetDefaultView
	resource-explorer-2:GetIndex
	resource-explorer-2:GetManagedView
	resource-explorer-2:GetResourceExplorerSetup
	resource-explorer-2:GetServiceIndex
	resource-explorer-2:GetServiceView
	resource-explorer-2:ListIndexes
	resource-explorer-2:ListIndexesForMembers
	resource-explorer-2:ListManagedViews
	resource-explorer-2:ListServiceIndexes
	resource-explorer-2:ListServiceViews

Prefixo do serviço	Ações do
	resource-explorer-2:ListStreamingAccessForServices
	resource-explorer-2:ListSupportedResourceTypes
	resource-explorer-2:ListViews
	resource-explorer-2:Search
	resource-explorer-2:UpdateIndexType
	resource-explorer-2:UpdateView

Prefixo do serviço	Ações do
resource-groups	resource-groups:CancelTagSyncTask resource-groups:GetAccountSettings resource-groups:GetGroup resource-groups:GetGroupConfiguration resource-groups:GetGroupQuery resource-groups:GetTagSyncTask resource-groups:GroupResources resource-groups:ListGroupResources resource-groups:ListGroupingStatuses resource-groups:ListGroups resource-groups:ListTagSyncTasks resource-groups:PutGroupConfiguration resource-groups:SearchResources resource-groups:StartTagSyncTask resource-groups:UngroupResources resource-groups:UpdateAccountSettings resource-groups:UpdateGroup resource-groups:UpdateGroupQuery

Prefixo do serviço	Ações do
robomaker	robomaker:BatchDeleteWorlds
	robomaker:BatchDescribeSimulationJob
	robomaker:CancelDeploymentJob
	robomaker:CancelSimulationJob
	robomaker:CancelSimulationJobBatch
	robomaker:CancelWorldExportJob
	robomaker:CancelWorldGenerationJob
	robomaker:CreateDeploymentJob
	robomaker:CreateFleet
	robomaker:CreateRobot
	robomaker:CreateRobotApplication
	robomaker:CreateRobotApplicationVersion
	robomaker:CreateSimulationApplication
	robomaker:CreateSimulationApplicationVersion
	robomaker:CreateSimulationJob
	robomaker:CreateWorldExportJob
	robomaker:CreateWorldGenerationJob
	robomaker:CreateWorldTemplate
	robomaker:DeleteFleet
	robomaker:DeleteRobot
	robomaker:DeleteRobotApplication

Prefixo do serviço	Ações do
	robomaker:DeleteSimulationApplication
	robomaker:DeleteWorldTemplate
	robomaker:DeregisterRobot
	robomaker:DescribeDeploymentJob
	robomaker:DescribeFleet
	robomaker:DescribeRobot
	robomaker:DescribeRobotApplication
	robomaker:DescribeSimulationApplication
	robomaker:DescribeSimulationJob
	robomaker:DescribeSimulationJobBatch
	robomaker:DescribeWorld
	robomaker:DescribeWorldExportJob
	robomaker:DescribeWorldGenerationJob
	robomaker:DescribeWorldTemplate
	robomaker:GetWorldTemplateBody
	robomaker:ListDeploymentJobs
	robomaker:ListFleets
	robomaker:ListRobotApplications
	robomaker:ListRobots
	robomaker:ListSimulationApplications
	robomaker:ListSimulationJobBatches

Prefixo do serviço	Ações do
	robomaker:ListSimulationJobs
	robomaker:ListWorldExportJobs
	robomaker:ListWorldGenerationJobs
	robomaker:ListWorldTemplates
	robomaker:ListWorlds
	robomaker:RegisterRobot
	robomaker:RestartSimulationJob
	robomaker:StartSimulationJobBatch
	robomaker:SyncDeploymentJob
	robomaker:UpdateRobotApplication
	robomaker:UpdateSimulationApplication
	robomaker:UpdateWorldTemplate

Prefixo do serviço	Ações do
rolesanywhere	rolesanywhere:CreateProfile rolesanywhere:CreateTrustAnchor rolesanywhere:DeleteAttributeMapping rolesanywhere:DeleteCrl rolesanywhere:DeleteProfile rolesanywhere:DeleteTrustAnchor rolesanywhere:DisableCrl rolesanywhere:DisableProfile rolesanywhere:DisableTrustAnchor rolesanywhere:EnableCrl rolesanywhere:EnableProfile rolesanywhere:EnableTrustAnchor rolesanywhere:GetCrl rolesanywhere:GetProfile rolesanywhere:GetSubject rolesanywhere:GetTrustAnchor rolesanywhere:ImportCrl rolesanywhere:ListCrls rolesanywhere:ListProfiles rolesanywhere:ListSubjects rolesanywhere:ListTrustAnchors

Prefixo do serviço	Ações do
	rolesanywhere:PutAttributeMapping rolesanywhere:PutNotificationSettings rolesanywhere:ResetNotificationSettings rolesanywhere:UpdateCrl rolesanywhere:UpdateProfile rolesanywhere:UpdateTrustAnchor

Prefixo do serviço	Ações do
route53	route53:ActivateKeySigningKey
	route53:AssociateVPCWithHostedZone
	route53:ChangeCidrCollection
	route53:ChangeResourceRecordSets
	route53:CreateCidrCollection
	route53:CreateHealthCheck
	route53:CreateHostedZone
	route53:CreateKeySigningKey
	route53:CreateQueryLoggingConfig
	route53:CreateReusableDelegationSet
	route53:CreateTrafficPolicy
	route53:CreateTrafficPolicyInstance
	route53:CreateTrafficPolicyVersion
	route53:CreateVPCAssociationAuthorization
	route53:DeactivateKeySigningKey
	route53:DeleteCidrCollection
	route53:DeleteHealthCheck
	route53:DeleteHostedZone
	route53:DeleteKeySigningKey
	route53:DeleteQueryLoggingConfig
	route53:DeleteReusableDelegationSet

Prefixo do serviço	Ações do
	route53:DeleteTrafficPolicy
	route53:DeleteTrafficPolicyInstance
	route53:DeleteVPCAssociationAuthorization
	route53:DisableHostedZoneDNSSEC
	route53:DisassociateVPCFromHostedZone
	route53:EnableHostedZoneDNSSEC
	route53:GetAccountLimit
	route53:GetChange
	route53:GetCheckerIpRanges
	route53:GetDNSSEC
	route53:GetGeoLocation
	route53:GetHealthCheck
	route53:GetHealthCheckCount
	route53:GetHealthCheckLastFailureReason
	route53:GetHealthCheckStatus
	route53:GetHostedZone
	route53:GetHostedZoneCount
	route53:GetHostedZoneLimit
	route53:GetQueryLoggingConfig
	route53:GetReusableDelegationSet
	route53:GetReusableDelegationSetLimit

Prefixo do serviço	Ações do
	route53:GetTrafficPolicy
	route53:GetTrafficPolicyInstance
	route53:GetTrafficPolicyInstanceCount
	route53:ListCidrBlocks
	route53:ListCidrCollections
	route53:ListCidrLocations
	route53:ListGeoLocations
	route53:ListHealthChecks
	route53:ListHostedZones
	route53:ListHostedZonesByName
	route53:ListHostedZonesByVPC
	route53:ListQueryLoggingConfigs
	route53:ListResourceRecordSets
	route53:ListReusableDelegationSets
	route53:ListTrafficPolicies
	route53:ListTrafficPolicyInstances
	route53:ListTrafficPolicyInstancesByHostedZone
	route53:ListTrafficPolicyInstancesByPolicy
	route53:ListTrafficPolicyVersions
	route53:ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

Prefixo do serviço	Ações do
	route53:UpdateHealthCheck
	route53:UpdateHostedZoneComment
	route53:UpdateTrafficPolicyComment
	route53:UpdateTrafficPolicyInstance

Prefixo do serviço	Ações do
route53-recovery-control-config	route53-recovery-control-config:CreateCluster route53-recovery-control-config:CreateControlPanel route53-recovery-control-config:CreateRoutingControl route53-recovery-control-config:CreateSafetyRule route53-recovery-control-config:DeleteCluster route53-recovery-control-config:DeleteControlPanel route53-recovery-control-config:DeleteRoutingControl route53-recovery-control-config:DeleteSafetyRule route53-recovery-control-config:DescribeCluster route53-recovery-control-config:DescribeControlPanel route53-recovery-control-config:DescribeRoutingControl route53-recovery-control-config:DescribeSafetyRule route53-recovery-control-config:GetResourcePolicy route53-recovery-control-config:ListAssociatedRoute53HealthChecks route53-recovery-control-config:ListClusters route53-recovery-control-config:ListControlPanels route53-recovery-control-config:ListRoutingControls route53-recovery-control-config:ListSafetyRules route53-recovery-control-config:UpdateCluster route53-recovery-control-config:UpdateControlPanel

Prefixo do serviço	Ações do
	route53-recovery-control-config:UpdateRoutingControl route53-recovery-control-config:UpdateSafetyRule

Prefixo do serviço	Ações do
route53-recovery-readiness	route53-recovery-readiness:CreateCell
	route53-recovery-readiness:CreateCrossAccountAuthorization
	route53-recovery-readiness:CreateReadinessCheck
	route53-recovery-readiness:CreateRecoveryGroup
	route53-recovery-readiness:CreateResourceSet
	route53-recovery-readiness:DeleteCell
	route53-recovery-readiness:DeleteCrossAccountAuthorization
	route53-recovery-readiness:DeleteReadinessCheck
	route53-recovery-readiness:DeleteRecoveryGroup
	route53-recovery-readiness:DeleteResourceSet
	route53-recovery-readiness:GetArchitectureRecommendations
	route53-recovery-readiness:GetCell
	route53-recovery-readiness:GetCellReadinessSummary
	route53-recovery-readiness:GetReadinessCheck
	route53-recovery-readiness:GetReadinessCheckResourceStatus
	route53-recovery-readiness:GetReadinessCheckStatus
	route53-recovery-readiness:GetRecoveryGroup
	route53-recovery-readiness:GetRecoveryGroupReadinessSummary
	route53-recovery-readiness:GetResourceSet
	route53-recovery-readiness:ListCells
	route53-recovery-readiness:ListCrossAccountAuthorizations

Prefixo do serviço	Ações do
	route53-recovery-readiness:ListReadinessChecks
	route53-recovery-readiness:ListRecoveryGroups
	route53-recovery-readiness:ListResourceSets
	route53-recovery-readiness:ListRules
	route53-recovery-readiness:UpdateCell
	route53-recovery-readiness:UpdateReadinessCheck
	route53-recovery-readiness:UpdateRecoveryGroup
	route53-recovery-readiness:UpdateResourceSet

Prefixo do serviço	Ações do
route53resolver	route53resolver:AssociateFirewallRuleGroup route53resolver:AssociateResolverEndpointIpAddress route53resolver:AssociateResolverQueryLogConfig route53resolver:AssociateResolverRule route53resolver:CreateFirewallDomainList route53resolver:CreateFirewallRule route53resolver:CreateFirewallRuleGroup route53resolver:CreateResolverEndpoint route53resolver:CreateResolverQueryLogConfig route53resolver:CreateResolverRule route53resolver:DeleteFirewallDomainList route53resolver:DeleteFirewallRule route53resolver:DeleteFirewallRuleGroup route53resolver:DeleteOutpostResolver route53resolver:DeleteResolverEndpoint route53resolver:DeleteResolverQueryLogConfig route53resolver:DeleteResolverRule route53resolver:DisassociateFirewallRuleGroup route53resolver:DisassociateResolverEndpointIpAddress route53resolver:DisassociateResolverQueryLogConfig route53resolver:DisassociateResolverRule

Prefixo do serviço	Ações do
	route53resolver:GetFirewallConfig route53resolver:GetFirewallDomainList route53resolver:GetFirewallRuleGroup route53resolver:GetFirewallRuleGroupAssociation route53resolver:GetFirewallRuleGroupPolicy route53resolver:GetOutpostResolver route53resolver:GetResolverConfig route53resolver:GetResolverDnssecConfig route53resolver:GetResolverEndpoint route53resolver:GetResolverQueryLogConfig route53resolver:GetResolverQueryLogConfigAssociation route53resolver:GetResolverQueryLogConfigPolicy route53resolver:GetResolverRule route53resolver:GetResolverRuleAssociation route53resolver:GetResolverRulePolicy route53resolver:ImportFirewallDomains route53resolver:ListFirewallConfigs route53resolver:ListFirewallDomainLists route53resolver:ListFirewallDomains route53resolver:ListFirewallRuleGroupAssociations route53resolver:ListFirewallRuleGroups

Prefixo do serviço	Ações do
	route53resolver:ListFirewallRuleTypes route53resolver:ListFirewallRules route53resolver:ListOutpostResolvers route53resolver:ListResolverConfigs route53resolver:ListResolverDnssecConfigs route53resolver:ListResolverEndpointIpAddresses route53resolver:ListResolverEndpoints route53resolver:ListResolverQueryLogConfigAssociations route53resolver:ListResolverQueryLogConfigs route53resolver:ListResolverRuleAssociations route53resolver:ListResolverRules route53resolver:PutFirewallRuleGroupPolicy route53resolver:PutResolverQueryLogConfigPolicy route53resolver:UpdateFirewallConfig route53resolver:UpdateFirewallDomains route53resolver:UpdateFirewallRule route53resolver:UpdateFirewallRuleGroupAssociation route53resolver:UpdateOutpostResolver route53resolver:UpdateResolverConfig route53resolver:UpdateResolverDnssecConfig route53resolver:UpdateResolverEndpoint

Prefixo do serviço	Ações do
	route53resolver:UpdateResolverRule
rum	rum:BatchCreateRumMetricDefinitions
	rum:BatchDeleteRumMetricDefinitions
	rum:BatchGetRumMetricDefinitions
	rum:CreateAppMonitor
	rum:DeleteAppMonitor
	rum:DeleteResourcePolicy
	rum:DeleteRumMetricsDestination
	rum:GetAppMonitor
	rum:GetAppMonitorData
	rum:GetResourcePolicy
	rum:ListAppMonitors
	rum:ListRumMetricsDestinations
	rum:PutResourcePolicy
	rum:PutRumMetricsDestination
	rum:UpdateAppMonitor
	rum:UpdateRumMetricDefinition

Prefixo do serviço	Ações do
s3	s3:AssociateAccessGrantsIdentityCenter s3:CreateAccessGrant s3:CreateAccessGrantsInstance s3:CreateAccessGrantsLocation s3:CreateAccessPoint s3:CreateAccessPointForObjectLambda s3:CreateBucket s3:CreateBucketMetadataTableConfiguration s3:CreateJob s3:CreateMultiRegionAccessPoint s3:DeleteAccessGrant s3:DeleteAccessGrantsInstance s3:DeleteAccessGrantsInstanceResourcePolicy s3:DeleteAccessGrantsLocation s3:DeleteAccessPoint s3:DeleteAccessPointForObjectLambda s3:DeleteAccessPointPolicy s3:DeleteAccessPointPolicyForObjectLambda s3:DeleteBucket s3:DeleteBucketMetadataTableConfiguration s3:DeleteBucketPolicy

Prefixo do serviço	Ações do
	s3:DeleteBucketWebsite s3:DeleteMultiRegionAccessPoint s3:DeleteStorageLensConfiguration s3:DescribeJob s3:DescribeMultiRegionAccessPointOperation s3:DissociateAccessGrantsIdentityCenter s3:GetAccelerateConfiguration s3:GetAccessGrant s3:GetAccessGrantsInstance s3:GetAccessGrantsInstanceForPrefix s3:GetAccessGrantsInstanceResourcePolicy s3:GetAccessGrantsLocation s3:GetAccessPoint s3:GetAccessPointConfigurationForObjectLambda s3:GetAccessPointForObjectLambda s3:GetAccessPointPolicy s3:GetAccessPointPolicyForObjectLambda s3:GetAccessPointPolicyStatus s3:GetAccessPointPolicyStatusForObjectLambda s3:GetAccountPublicAccessBlock s3:GetAnalyticsConfiguration

Prefixo do serviço	Ações do
	s3:GetBucketAbac s3:GetBucketAcl s3:GetBucketCORS s3:GetBucketLocation s3:GetBucketLogging s3:GetBucketNotification s3:GetBucketObjectLockConfiguration s3:GetBucketOwnershipControls s3:GetBucketPolicy s3:GetBucketPolicyStatus s3:GetBucketPublicAccessBlock s3:GetBucketRequestPayment s3:GetBucketVersioning s3:GetBucketWebsite s3:GetDataAccess s3:GetEncryptionConfiguration s3:GetIntelligentTieringConfiguration s3:GetInventoryConfiguration s3:GetLifecycleConfiguration s3:GetMetricsConfiguration s3:GetMultiRegionAccessPoint

Prefixo do serviço	Ações do
	s3:GetMultiRegionAccessPointPolicy s3:GetMultiRegionAccessPointPolicyStatus s3:GetMultiRegionAccessPointRoutes s3:GetReplicationConfiguration s3:GetStorageLensConfiguration s3:GetStorageLensDashboard s3:ListAccessGrants s3:ListAccessGrantsInstances s3:ListAccessGrantsLocations s3:ListAccessPoints s3:ListAccessPointsForObjectLambda s3:ListAllMyBuckets s3:ListBucketMultipartUploads s3:ListCallerAccessGrants s3:ListJobs s3:ListMultiRegionAccessPoints s3:ListStorageLensConfigurations s3:PutAccelerateConfiguration s3:PutAccessGrantsInstanceResourcePolicy s3:PutAccessPointConfigurationForObjectLambda s3:PutAccessPointPolicy

Prefixo do serviço	Ações do
	s3:PutAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:PutAnalyticsConfiguration s3:PutBucketAbac s3:PutBucketAcl s3:PutBucketCORS s3:PutBucketLogging s3:PutBucketNotification s3:PutBucketObjectLockConfiguration s3:PutBucketOwnershipControls s3:PutBucketPolicy s3:PutBucketPublicAccessBlock s3:PutBucketRequestPayment s3:PutBucketVersioning s3:PutBucketWebsite s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutMetricsConfiguration s3:PutMultiRegionAccessPointPolicy

Prefixo do serviço	Ações do
	s3:PutReplicationConfiguration s3:PutStorageLensConfiguration s3:SubmitMultiRegionAccessPointRoutes s3:UpdateAccessGrantsLocation s3:UpdateBucketMetadataAnnotationTableConfiguration s3:UpdateBucketMetadataJournalTableConfiguration s3:UpdateJobPriority s3:UpdateJobStatus
s3-outposts	s3-outposts:CreateEndpoint s3-outposts:DeleteEndpoint s3-outposts:ListEndpoints s3-outposts:ListOutpostsWithS3 s3-outposts:ListSharedEndpoints

Prefixo do serviço	Ações do
sagemaker-geospatial	sagemaker-geospatial:DeleteEarthObservationJob sagemaker-geospatial:DeleteVectorEnrichmentJob sagemaker-geospatial:ExportEarthObservationJob sagemaker-geospatial:ExportVectorEnrichmentJob sagemaker-geospatial:GetEarthObservationJob sagemaker-geospatial:GetRasterDataCollection sagemaker-geospatial:GetTile sagemaker-geospatial:GetVectorEnrichmentJob sagemaker-geospatial:ListEarthObservationJobs sagemaker-geospatial:ListRasterDataCollections sagemaker-geospatial:ListVectorEnrichmentJobs sagemaker-geospatial:SearchRasterDataCollection sagemaker-geospatial:StartEarthObservationJob sagemaker-geospatial:StartVectorEnrichmentJob sagemaker-geospatial:StopEarthObservationJob sagemaker-geospatial:StopVectorEnrichmentJob

Prefixo do serviço	Ações do
savingsplans	savingsplans:CreateSavingsPlan
	savingsplans:DeleteQueuedSavingsPlan
	savingsplans:DescribeSavingsPlanRates
	savingsplans:DescribeSavingsPlans
	savingsplans:DescribeSavingsPlansOfferingRates
	savingsplans:DescribeSavingsPlansOfferings
	savingsplans:ReturnSavingsPlan

Prefixo do serviço	Ações do
schemas	schemas:CreateDiscoverer schemas:CreateRegistry schemas:CreateSchema schemas:DeleteDiscoverer schemas:DeleteRegistry schemas:DeleteResourcePolicy schemas:DeleteSchema schemas:DeleteSchemaVersion schemas:DescribeCodeBinding schemas:DescribeDiscoverer schemas:DescribeRegistry schemas:DescribeSchema schemas:ExportSchema schemas:GetCodeBindingSource schemas:GetDiscoveredSchema schemas:GetResourcePolicy schemas:ListDiscoverers schemas:ListRegistries schemas:ListSchemaVersions schemas:ListSchemas schemas:PutCodeBinding

Prefixo do serviço	Ações do
	schemas:PutResourcePolicy schemas:SearchSchemas schemas:StartDiscoverer schemas:StopDiscoverer schemas:UpdateDiscoverer schemas:UpdateRegistry schemas:UpdateSchema
sdb	sdb:CreateDomain sdb:DeleteDomain sdb:DomainMetadata sdb:ListDomains

Prefixo do serviço	Ações do
secretsmanager	secretsmanager:CancelRotateSecret secretsmanager:CreateSecret secretsmanager:DeleteResourcePolicy secretsmanager:DeleteSecret secretsmanager:DescribeSecret secretsmanager:GetRandomPassword secretsmanager:GetResourcePolicy secretsmanager:GetSecretValue secretsmanager:ListSecretVersionIds secretsmanager:ListSecrets secretsmanager:PutResourcePolicy secretsmanager:PutSecretValue secretsmanager:RemoveRegionsFromReplication secretsmanager:ReplicateSecretToRegions secretsmanager:RestoreSecret secretsmanager:RotateSecret secretsmanager:StopReplicationToReplica secretsmanager:UpdateSecret secretsmanager:ValidateResourcePolicy

Prefixo do serviço	Ações do
securityhub	securityhub:AcceptAdministratorInvitation
	securityhub:AcceptInvitation
	securityhub:BatchDeleteAutomationRules
	securityhub:BatchDisableStandards
	securityhub:BatchEnableStandards
	securityhub:BatchGetAutomationRules
	securityhub:BatchGetConfigurationPolicyAssociations
	securityhub:BatchGetEnabledRegionsV2
	securityhub:BatchGetSecurityControls
	securityhub:BatchGetStandardsControlAssociations
	securityhub:BatchImportFindings
	securityhub:BatchUpdateAutomationRules
	securityhub:BatchUpdateFindings
	securityhub:BatchUpdateStandardsControlAssociations
	securityhub:ConnectorRegistrationsV2
	securityhub:CreateActionTarget
	securityhub:CreateAggregatorV2
	securityhub:CreateAutomationRule
	securityhub:CreateAutomationRuleV2
	securityhub:CreateConfigurationPolicy
	securityhub:CreateConnector

Prefixo do serviço	Ações do
	securityhub:CreateConnectorV2
	securityhub:CreateFindingAggregator
	securityhub:CreateInsight
	securityhub:CreateMembers
	securityhub:CreateTicketV2
	securityhub:DeclineInvitations
	securityhub:DeleteActionTarget
	securityhub:DeleteAggregatorV2
	securityhub:DeleteAutomationRuleV2
	securityhub:DeleteConfigurationPolicy
	securityhub:DeleteConnector
	securityhub:DeleteConnectorV2
	securityhub:DeleteFindingAggregator
	securityhub:DeleteInsight
	securityhub:DeleteInvitations
	securityhub:DeleteMembers
	securityhub:DescribeActionTargets
	securityhub:DescribeHub
	securityhub:DescribeOrganizationConfiguration
	securityhub:DescribeProducts
	securityhub:DescribeSecurityHubV2

Prefixo do serviço	Ações do
	securityhub:DescribeStandards
	securityhub:DisableImportFindingsForProduct
	securityhub:DisableOrganizationAdminAccount
	securityhub:DisableSecurityHub
	securityhub:DisableSecurityHubFeatureV2
	securityhub:DisableSecurityHubV2
	securityhub:DisassociateFromAdministratorAccount
	securityhub:DisassociateFromMasterAccount
	securityhub:DisassociateMembers
	securityhub:EnableImportFindingsForProduct
	securityhub:EnableOrganizationAdminAccount
	securityhub:EnableSecurityHub
	securityhub:EnableSecurityHubFeatureV2
	securityhub:GetAdministratorAccount
	securityhub:GetAggregatorV2
	securityhub:GetAutomationRuleV2
	securityhub:GetConfigurationPolicy
	securityhub:GetConfigurationPolicyAssociation
	securityhub:GetConnector
	securityhub:GetConnectorV2
	securityhub:GetCoverageStatisticsV2

Prefixo do serviço	Ações do
	securityhub:GetEnabledStandards
	securityhub:GetFindingAggregator
	securityhub:GetFindingHistory
	securityhub:GetFindings
	securityhub:GetInsightResults
	securityhub:GetInsights
	securityhub:GetInvitationsCount
	securityhub:GetMasterAccount
	securityhub:GetMembers
	securityhub:GetSecurityControlDefinition
	securityhub:InviteMembers
	securityhub:ListAggregatorsV2
	securityhub:ListAutomationRules
	securityhub:ListAutomationRulesV2
	securityhub:ListConfigurationPolicies
	securityhub:ListConfigurationPolicyAssociations
	securityhub:ListConnectors
	securityhub:ListConnectorsV2
	securityhub:ListEnabledProductsForImport
	securityhub:ListFindingAggregators
	securityhub:ListInvitations

Prefixo do serviço	Ações do
	securityhub:ListMembers
	securityhub:ListOrganizationAdminAccounts
	securityhub:ListSecurityControlDefinitions
	securityhub:ListStandardsControlAssociations
	securityhub:StartConfigurationPolicyAssociation
	securityhub:StartConfigurationPolicyDisassociation
	securityhub:UpdateActionTarget
	securityhub:UpdateAggregatorV2
	securityhub:UpdateAutomationRuleV2
	securityhub:UpdateConfigurationPolicy
	securityhub:UpdateConnector
	securityhub:UpdateConnectorV2
	securityhub:UpdateFindingAggregator
	securityhub:UpdateFindings
	securityhub:UpdateInsight
	securityhub:UpdateOrganizationConfiguration
	securityhub:UpdateSecurityControl
	securityhub:UpdateSecurityHubConfiguration

Prefixo do serviço	Ações do
securitylake	securitylake:CreateAwsLogSource securitylake:CreateCustomLogSource securitylake:CreateDataLakeExceptionSubscription securitylake:CreateDataLakeOrganizationConfiguration securitylake:CreateSubscriber securitylake:CreateSubscriberNotification securitylake:DeleteAwsLogSource securitylake:DeleteCustomLogSource securitylake:DeleteDataLakeExceptionSubscription securitylake:DeleteDataLakeOrganizationConfiguration securitylake:DeleteSubscriber securitylake:DeleteSubscriberNotification securitylake:DeregisterDataLakeDelegatedAdministrator securitylake:GetDataLakeExceptionSubscription securitylake:GetDataLakeOrganizationConfiguration securitylake:GetDataLakeSources securitylake:GetSubscriber securitylake:ListDataLakes securitylake:ListLogSources securitylake:ListSubscribers securitylake:RegisterDataLakeDelegatedAdministrator

Prefixo do serviço	Ações do
	securitylake:UpdateDataLakeExceptionSubscription securitylake:UpdateSubscriber securitylake:UpdateSubscriberNotification
serverlessrepo	serverlessrepo:CreateApplication serverlessrepo:CreateApplicationVersion serverlessrepo:CreateCloudFormationChangeSet serverlessrepo:CreateCloudFormationTemplate serverlessrepo:DeleteApplication serverlessrepo:GetApplication serverlessrepo:GetApplicationPolicy serverlessrepo:GetCloudFormationTemplate serverlessrepo:ListApplicationDependencies serverlessrepo:ListApplicationVersions serverlessrepo:ListApplications serverlessrepo:PutApplicationPolicy serverlessrepo:UnshareApplication serverlessrepo:UpdateApplication

Prefixo do serviço	Ações do
servicecatalog	servicecatalog:AcceptPortfolioShare servicecatalog:AssociateBudgetWithResource servicecatalog:AssociatePrincipalWithPortfolio servicecatalog:AssociateProductWithPortfolio servicecatalog:AssociateServiceActionWithProvisioningArtifact servicecatalog:BatchAssociateServiceActionWithProvisioningArtifact servicecatalog:BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog:CopyProduct servicecatalog:CreateAttributeGroup servicecatalog:CreateConstraint servicecatalog:CreatePortfolio servicecatalog:CreatePortfolioShare servicecatalog:CreateProduct servicecatalog:CreateProvisionedProductPlan servicecatalog:CreateProvisioningArtifact servicecatalog:CreateServiceAction servicecatalog:DeleteAttributeGroup servicecatalog:DeleteConstraint servicecatalog:DeletePortfolio servicecatalog:DeletePortfolioShare

Prefixo do serviço	Ações do
	servicecatalog:DeleteProduct servicecatalog:DeleteProvisionedProductPlan servicecatalog:DeleteProvisioningArtifact servicecatalog:DeleteServiceAction servicecatalog:DescribeConstraint servicecatalog:DescribeCopyProductStatus servicecatalog:DescribePortfolio servicecatalog:DescribePortfolioShareStatus servicecatalog:DescribePortfolioShares servicecatalog:DescribeProduct servicecatalog:DescribeProductAsAdmin servicecatalog:DescribeProductView servicecatalog:DescribeProvisionedProduct servicecatalog:DescribeProvisionedProductPlan servicecatalog:DescribeProvisioningArtifact servicecatalog:DescribeProvisioningParameters servicecatalog:DescribeRecord servicecatalog:DescribeServiceAction servicecatalog:DescribeServiceActionExecutionParameters servicecatalog:DisableAWSOrganizationsAccess servicecatalog:DisassociateBudgetFromResource

Prefixo do serviço	Ações do
	servicecatalog:DisassociatePrincipalFromPortfolio servicecatalog:DisassociateProductFromPortfolio servicecatalog:DisassociateServiceActionFromProvisioningArtifact servicecatalog:EnableAWSOrganizationsAccess servicecatalog:ExecuteProvisionedProductPlan servicecatalog:ExecuteProvisionedProductServiceAction servicecatalog:GetAWSOrganizationsAccessStatus servicecatalog:GetProvisionedProductOutputs servicecatalog:ImportAsProvisionedProduct servicecatalog:ListAcceptedPortfolioShares servicecatalog:ListAttributeGroups servicecatalog:ListBudgetsForResource servicecatalog:ListConstraintsForPortfolio servicecatalog:ListLaunchPaths servicecatalog:ListOrganizationPortfolioAccess servicecatalog:ListPortfolioAccess servicecatalog:ListPortfolios servicecatalog:ListPortfoliosForProduct servicecatalog:ListPrincipalsForPortfolio servicecatalog:ListProvisionedProductPlans servicecatalog:ListProvisioningArtifacts

Prefixo do serviço	Ações do
	servicecatalog:ListProvisioningArtifactsForServiceAction servicecatalog:ListRecordHistory servicecatalog:ListServiceActions servicecatalog:ListServiceActionsForProvisioningArtifact servicecatalog:ListStackInstancesForProvisionedProduct servicecatalog:NotifyProvisionProductEngineWorkflowResult servicecatalog:NotifyTerminateProvisionedProductEngine WorkflowResult servicecatalog:NotifyUpdateProvisionedProductEngineWor kflowResult servicecatalog:ProvisionProduct servicecatalog:RejectPortfolioShare servicecatalog:ScanProvisionedProducts servicecatalog:SearchProducts servicecatalog:SearchProductsAsAdmin servicecatalog:SearchProvisionedProducts servicecatalog:TerminateProvisionedProduct servicecatalog:UpdateConstraint servicecatalog:UpdatePortfolio servicecatalog:UpdatePortfolioShare servicecatalog:UpdateProduct servicecatalog:UpdateProvisionedProduct

Prefixo do serviço	Ações do
	servicecatalog:UpdateProvisionedProductProperties servicecatalog:UpdateProvisioningArtifact servicecatalog:UpdateServiceAction

Prefixo do serviço	Ações do
servicediscovery	servicediscovery:CreateHttpNamespace
	servicediscovery:CreatePrivateDnsNamespace
	servicediscovery:CreatePublicDnsNamespace
	servicediscovery:CreateService
	servicediscovery:DeleteNamespace
	servicediscovery:DeleteService
	servicediscovery:DeleteServiceAttributes
	servicediscovery:DeregisterInstance
	servicediscovery:GetInstance
	servicediscovery:GetInstancesHealthStatus
	servicediscovery:GetNamespace
	servicediscovery:GetOperation
	servicediscovery:GetService
	servicediscovery:ListInstances
	servicediscovery:ListNamespaces
	servicediscovery:ListOperations
	servicediscovery:ListServices
	servicediscovery:RegisterInstance
	servicediscovery:UpdateHttpNamespace
	servicediscovery:UpdateInstanceCustomHealthStatus
	servicediscovery:UpdatePrivateDnsNamespace

Prefixo do serviço	Ações do
	servicediscovery:UpdatePublicDnsNamespace servicediscovery:UpdateService servicediscovery:UpdateServiceAttributes

Prefixo do serviço	Ações do
servicequotas	servicequotas:AssociateServiceQuotaTemplate servicequotas:CreateSupportCase servicequotas:DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas:DisassociateServiceQuotaTemplate servicequotas:GetAWSDefaultServiceQuota servicequotas:GetAssociationForServiceQuotaTemplate servicequotas:GetAutoManagementConfiguration servicequotas:GetQuotaUtilizationReport servicequotas:GetRequestedServiceQuotaChange servicequotas:GetServiceQuota servicequotas:GetServiceQuotaIncreaseRequestFromTemplate servicequotas:ListAWSDefaultServiceQuotas servicequotas:ListRequestedServiceQuotaChangeHistory servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota servicequotas:ListServiceQuotaIncreaseRequestsInTemplate servicequotas:ListServiceQuotas servicequotas:ListServices servicequotas:PutServiceQuotaIncreaseRequestIntoTemplate servicequotas:RequestServiceQuotaIncrease servicequotas:StartAutoManagement servicequotas:StartQuotaUtilizationReport

Prefixo do serviço	Ações do
	servicequotas:StopAutoManagement servicequotas:UpdateAutoManagement

Prefixo do serviço	Ações do
ses	ses:BatchGetMetricData
	ses:CloneReceiptRuleSet
	ses:CreateAddonInstance
	ses:CreateAddonSubscription
	ses:CreateAddressList
	ses:CreateAddressListImportJob
	ses:CreateArchive
	ses:CreateConfigurationSet
	ses:CreateConfigurationSetEventDestination
	ses:CreateConfigurationSetTrackingOptions
	ses:CreateContact
	ses:CreateContactList
	ses:CreateCustomVerificationEmailTemplate
	ses:CreateDedicatedIpPool
	ses:CreateDeliverabilityTestReport
	ses:CreateEmailIdentity
	ses:CreateEmailIdentityPolicy
	ses:CreateEmailTemplate
	ses:CreateImportJob
	ses:CreateIngressPoint
	ses:CreateMultiRegionEndpoint

Prefixo do serviço	Ações do
	ses:CreateReceiptFilter
	ses:CreateReceiptRule
	ses:CreateReceiptRuleSet
	ses:CreateRelay
	ses:CreateRuleSet
	ses:CreateTemplate
	ses:CreateTenant
	ses:CreateTenantResourceAssociation
	ses:CreateTrafficPolicy
	ses:DeleteAddonInstance
	ses:DeleteAddonSubscription
	ses:DeleteAddressList
	ses:DeleteArchive
	ses:DeleteConfigurationSet
	ses:DeleteConfigurationSetEventDestination
	ses:DeleteConfigurationSetTrackingOptions
	ses:DeleteContact
	ses:DeleteContactList
	ses:DeleteCustomVerificationEmailTemplate
	ses:DeleteDedicatedIpPool
	ses:DeleteEmailIdentity

Prefixo do serviço	Ações do
	ses:DeleteEmailIdentityPolicy
	ses:DeleteEmailTemplate
	ses:DeleteIdentity
	ses:DeleteIdentityPolicy
	ses:DeleteIngressPoint
	ses:DeleteMultiRegionEndpoint
	ses:DeleteReceiptFilter
	ses:DeleteReceiptRule
	ses:DeleteReceiptRuleSet
	ses:DeleteRelay
	ses:DeleteRuleSet
	ses:DeleteSuppressedDestination
	ses:DeleteTemplate
	ses:DeleteTenant
	ses:DeleteTenantResourceAssociation
	ses:DeleteTrafficPolicy
	ses:DeleteVerifiedEmailAddress
	ses:DeregisterMemberFromAddressList
	ses:DescribeActiveReceiptRuleSet
	ses:DescribeConfigurationSet
	ses:DescribeReceiptRule

Prefixo do serviço	Ações do
	ses:DescribeReceiptRuleSet
	ses:GetAccount
	ses:GetAccountSendingEnabled
	ses:GetAddonInstance
	ses:GetAddonSubscription
	ses:GetAddressList
	ses:GetArchive
	ses:GetArchiveExport
	ses:GetArchiveMessage
	ses:GetArchiveMessageContent
	ses:GetArchiveSearch
	ses:GetArchiveSearchResults
	ses:GetBlacklistReports
	ses:GetConfigurationSet
	ses:GetConfigurationSetEventDestinations
	ses:GetContact
	ses:GetContactList
	ses:GetCustomVerificationEmailTemplate
	ses:GetDedicatedIp
	ses:GetDedicatedIpPool
	ses:GetDedicatedIps

Prefixo do serviço	Ações do
	ses:GetDeliverabilityDashboardOptions
	ses:GetDeliverabilityTestReport
	ses:GetDomainDeliverabilityCampaign
	ses:GetDomainStatisticsReport
	ses:GetEmailAddressInsights
	ses:GetEmailIdentity
	ses:GetEmailIdentityPolicies
	ses:GetEmailTemplate
	ses:GetIdentityDkimAttributes
	ses:GetIdentityMailFromDomainAttributes
	ses:GetIdentityNotificationAttributes
	ses:GetIdentityPolicies
	ses:GetIdentityVerificationAttributes
	ses:GetImportJob
	ses:GetIngressPoint
	ses:GetMemberOfAddressList
	ses:GetMessageInsights
	ses:GetMultiRegionEndpoint
	ses:GetRelay
	ses:GetRuleSet
	ses:GetSendQuota

Prefixo do serviço	Ações do
	ses:GetSendStatistics
	ses:GetSuppressedDestination
	ses:GetTemplate
	ses:GetTenant
	ses:GetTrafficPolicy
	ses:ListAddonInstances
	ses:ListAddonSubscriptions
	ses:ListAddressListImportJobs
	ses:ListAddressLists
	ses:ListArchiveExports
	ses:ListArchiveSearches
	ses:ListArchives
	ses:ListConfigurationSets
	ses:ListContactLists
	ses:ListContacts
	ses:ListCustomVerificationEmailTemplates
	ses:ListDedicatedIpPools
	ses:ListDeliverabilityTestReports
	ses:ListDomainDeliverabilityCampaigns
	ses:ListEmailIdentities
	ses:ListEmailTemplates

Prefixo do serviço	Ações do
	ses:ListExportJobs
	ses:ListIdentities
	ses:ListIdentityPolicies
	ses:ListImportJobs
	ses:ListIngressPoints
	ses:ListMembersOfAddressList
	ses:ListMultiRegionEndpoints
	ses:ListReceiptFilters
	ses:ListReceiptRuleSets
	ses:ListRecommendations
	ses:ListRelays
	ses:ListReputationEntities
	ses:ListResourceTenants
	ses:ListRuleSets
	ses:ListSuppressedDestinations
	ses:ListTemplates
	ses:ListTenantResources
	ses:ListTenants
	ses:ListTrafficPolicies
	ses:ListVerifiedEmailAddresses
	ses:PutAccountDedicatedIpWarmupAttributes

Prefixo do serviço	Ações do
	ses:PutAccountDetails
	ses:PutAccountSendingAttributes
	ses:PutAccountSuppressionAttributes
	ses:PutAccountVdmAttributes
	ses:PutConfigurationSetArchivingOptions
	ses:PutConfigurationSetDeliveryOptions
	ses:PutConfigurationSetReputationOptions
	ses:PutConfigurationSetSendingOptions
	ses:PutConfigurationSetSuppressionOptions
	ses:PutConfigurationSetTrackingOptions
	ses:PutConfigurationSetVdmOptions
	ses:PutDedicatedIpInPool
	ses:PutDedicatedIpPoolScalingAttributes
	ses:PutDedicatedIpWarmupAttributes
	ses:PutDeliverabilityDashboardOption
	ses:PutEmailIdentityConfigurationSetAttributes
	ses:PutEmailIdentityDkimAttributes
	ses:PutEmailIdentityDkimSigningAttributes
	ses:PutEmailIdentityFeedbackAttributes
	ses:PutEmailIdentityMailFromAttributes
	ses:PutIdentityPolicy

Prefixo do serviço	Ações do
	ses:PutSuppressedDestination
	ses:PutTenantSuppressionAttributes
	ses:RegisterMemberToAddressList
	ses:ReorderReceiptRuleSet
	ses:SendBounce
	ses:SendCustomVerificationEmail
	ses:SetActiveReceiptRuleSet
	ses:SetIdentityDkimEnabled
	ses:SetIdentityFeedbackForwardingEnabled
	ses:SetIdentityHeadersInNotificationsEnabled
	ses:SetIdentityMailFromDomain
	ses:SetIdentityNotificationTopic
	ses:SetReceiptRulePosition
	ses:StartArchiveExport
	ses:StartArchiveSearch
	ses:StopArchiveExport
	ses:StopArchiveSearch
	ses:TestRenderEmailTemplate
	ses:TestRenderTemplate
	ses:UpdateAccountSendingEnabled
	ses:UpdateArchive

Prefixo do serviço	Ações do
	ses:UpdateConfigurationSetEventDestination
	ses:UpdateConfigurationSetReputationMetricsEnabled
	ses:UpdateConfigurationSetSendingEnabled
	ses:UpdateConfigurationSetTrackingOptions
	ses:UpdateContact
	ses:UpdateContactList
	ses:UpdateCustomVerificationEmailTemplate
	ses:UpdateEmailIdentityPolicy
	ses:UpdateEmailTemplate
	ses:UpdateIngressPoint
	ses:UpdateReceiptRule
	ses:UpdateRelay
	ses:UpdateRuleSet
	ses:UpdateTemplate
	ses:UpdateTrafficPolicy
	ses:VerifyDomainDkim
	ses:VerifyDomainIdentity
	ses:VerifyEmailAddress
	ses:VerifyEmailIdentity

Prefixo do serviço	Ações do
shield	shield:AssociateDRTLogBucket shield:AssociateHealthCheck shield:AssociateProactiveEngagementDetails shield:CreateProtection shield:CreateProtectionGroup shield:CreateSubscription shield:DeleteProtection shield:DeleteProtectionGroup shield:DeleteSubscription shield:DescribeAttack shield:DescribeAttackStatistics shield:DescribeDRTAccess shield:DescribeEmergencyContactSettings shield:DescribeProtection shield:DescribeProtectionGroup shield:DescribeSubscription shield:DisableApplicationLayerAutomaticResponse shield:DisableProactiveEngagement shield:DisassociateDRTLogBucket shield:DisassociateDRTRole shield:DisassociateHealthCheck

Prefixo do serviço	Ações do
	shield:EnableApplicationLayerAutomaticResponse shield:EnableProactiveEngagement shield:GetSubscriptionState shield:ListAttacks shield:ListProtectionGroups shield:ListProtections shield:ListResourcesInProtectionGroup shield:UpdateApplicationLayerAutomaticResponse shield:UpdateEmergencyContactSettings shield:UpdateProtectionGroup shield:UpdateSubscription

Prefixo do serviço	Ações do
signer	signer:AddProfilePermission
	signer:CancelSigningProfile
	signer:DescribeSigningJob
	signer:GetSigningPlatform
	signer:GetSigningProfile
	signer:ListProfilePermissions
	signer:ListSigningJobs
	signer:ListSigningPlatforms
	signer:ListSigningProfiles
	signer:PutSigningProfile
	signer:RemoveProfilePermission
	signer:RevokeSignature
	signer:RevokeSigningProfile
	signer:SignPayload
	signer:StartSigningJob

Prefixo do serviço	Ações do
simspaceweaver	simspaceweaver:CreateSnapshot simspaceweaver:DeleteApp simspaceweaver:DeleteSimulation simspaceweaver:DescribeApp simspaceweaver:DescribeSimulation simspaceweaver:ListApps simspaceweaver:ListSimulations simspaceweaver:StartApp simspaceweaver:StartClock simspaceweaver:StartSimulation simspaceweaver:StopApp simspaceweaver:StopClock simspaceweaver:StopSimulation

Prefixo do serviço	Ações do
sms	sms:CreateApp
	sms:CreateReplicationJob
	sms:DeleteApp
	sms:DeleteAppLaunchConfiguration
	sms:DeleteAppReplicationConfiguration
	sms:DeleteAppValidationConfiguration
	sms:DeleteReplicationJob
	sms:DeleteServerCatalog
	sms:DisassociateConnector
	sms:GenerateChangeSet
	sms:GenerateTemplate
	sms:GetApp
	sms:GetAppLaunchConfiguration
	sms:GetAppReplicationConfiguration
	sms:GetAppValidationConfiguration
	sms:GetAppValidationOutput
	sms:GetConnectors
	sms:GetReplicationJobs
	sms:GetReplicationRuns
	sms:GetServers
	sms:ImportAppCatalog

Prefixo do serviço	Ações do
	sms:ImportServerCatalog
	sms:LaunchApp
	sms:ListApps
	sms:NotifyAppValidationOutput
	sms:PutAppLaunchConfiguration
	sms:PutAppReplicationConfiguration
	sms:PutAppValidationConfiguration
	sms:StartAppReplication
	sms:StartOnDemandAppReplication
	sms:StartOnDemandReplicationRun
	sms:StopAppReplication
	sms:TerminateApp
	sms:UpdateApp
	sms:UpdateReplicationJob

Prefixo do serviço	Ações do
sms-voice	sms-voice:AssociateProtectConfiguration sms-voice:CreateConfigurationSet sms-voice:CreateConfigurationSetEventDestination sms-voice:CreateEventDestination sms-voice:CreateOptOutList sms-voice:CreatePool sms-voice:CreateProtectConfiguration sms-voice:CreateRegistration sms-voice:CreateRegistrationAssociation sms-voice:CreateRegistrationAttachment sms-voice:CreateRegistrationVersion sms-voice:CreateVerifiedDestinationNumber sms-voice:DeleteAccountDefaultProtectConfiguration sms-voice:DeleteConfigurationSet sms-voice:DeleteConfigurationSetEventDestination sms-voice:DeleteDefaultMessageType sms-voice:DeleteDefaultSenderId sms-voice:DeleteEventDestination sms-voice:DeleteKeyword sms-voice:DeleteMediaMessageSpendLimitOverride sms-voice:DeleteOptOutList

Prefixo do serviço	Ações do
	sms-voice:DeleteOptedOutNumber sms-voice:DeletePool sms-voice:DeleteProtectConfiguration sms-voice:DeleteProtectConfigurationRuleSetNumberOverride sms-voice:DeleteRegistration sms-voice:DeleteRegistrationAttachment sms-voice:DeleteResourcePolicy sms-voice:DeleteTextMessageSpendLimitOverride sms-voice:DeleteVerifiedDestinationNumber sms-voice:DeleteVoiceMessageSpendLimitOverride sms-voice:DescribeAccountAttributes sms-voice:DescribeAccountLimits sms-voice:DescribeConfigurationSets sms-voice:DescribeKeywords sms-voice:DescribeOptOutLists sms-voice:DescribeOptedOutNumbers sms-voice:DescribePhoneNumbers sms-voice:DescribePools sms-voice:DescribeProtectConfigurations sms-voice:DescribeRegistrationAttachments sms-voice:DescribeRegistrationFieldDefinitions

Prefixo do serviço	Ações do
	sms-voice:DescribeRegistrationFieldValues sms-voice:DescribeRegistrationSectionDefinitions sms-voice:DescribeRegistrationTypeDefinitions sms-voice:DescribeRegistrationVersions sms-voice:DescribeRegistrations sms-voice:DescribeSenderIds sms-voice:DescribeSpendLimits sms-voice:DescribeVerifiedDestinationNumbers sms-voice:DisassociateOriginationIdentity sms-voice:DisassociateProtectConfiguration sms-voice:DiscardRegistrationVersion sms-voice:GetConfigurationSetEventDestinations sms-voice:GetProtectConfigurationCountryRuleSet sms-voice:GetResourcePolicy sms-voice:ListConfigurationSets sms-voice:ListPoolOriginationIdentities sms-voice:ListProtectConfigurationRuleSetNumberOverrides sms-voice:ListRegistrationAssociations sms-voice:PutKeyword sms-voice:PutOptedOutNumber sms-voice:PutProtectConfigurationRuleSetNumberOverride

Prefixo do serviço	Ações do
	sms-voice:PutResourcePolicy sms-voice:ReleasePhoneNumber sms-voice:ReleaseSenderId sms-voice:RequestPhoneNumber sms-voice:RequestSenderId sms-voice:SendDestinationNumberVerificationCode sms-voice:SetAccountDefaultProtectConfiguration sms-voice:SetDefaultMessageFeedbackEnabled sms-voice:SetDefaultMessageType sms-voice:SetDefaultSenderId sms-voice:SetMediaMessageSpendLimitOverride sms-voice:SetTextMessageSpendLimitOverride sms-voice:SetVoiceMessageSpendLimitOverride sms-voice:SubmitRegistrationVersion sms-voice:UpdateConfigurationSetEventDestination sms-voice:UpdateEventDestination sms-voice:UpdatePhoneNumber sms-voice:UpdatePool sms-voice:UpdateProtectConfiguration sms-voice:UpdateProtectConfigurationCountryRuleSet sms-voice:UpdateSenderId

Prefixo do serviço	Ações do
snowball	snowball:CancelCluster
	snowball:CancelJob
	snowball:CreateAddress
	snowball:CreateCluster
	snowball:CreateJob
	snowball:CreateLongTermPricing
	snowball:CreateReturnShippingLabel
	snowball:DescribeAddress
	snowball:DescribeAddresses
	snowball:DescribeCluster
	snowball:DescribeJob
	snowball:DescribeReturnShippingLabel
	snowball:GetJobManifest
	snowball:GetJobUnlockCode
	snowball:GetSnowballUsage
	snowball:GetSoftwareUpdates
	snowball:ListClusterJobs
	snowball:ListClusters
	snowball:ListCompatibleImages
	snowball:ListJobs
	snowball:ListLongTermPricing

Prefixo do serviço	Ações do
	snowball:ListPickupLocations snowball:ListServiceVersions snowball:UpdateCluster snowball:UpdateJob snowball:UpdateJobShipmentState snowball:UpdateLongTermPricing
sqs	sqs:AddPermission sqs:CancelMessageMoveTask sqs:CreateQueue sqs:DeleteQueue sqs:PurgeQueue sqs:RemovePermission sqs:SetQueueAttributes

Prefixo do serviço	Ações do
ssm	ssm:AssociateOpsItemRelatedItem
	ssm:CancelCommand
	ssm:CancelMaintenanceWindowExecution
	ssm:CreateActivation
	ssm:CreateAssociation
	ssm:CreateAssociationBatch
	ssm:CreateDocument
	ssm:CreateMaintenanceWindow
	ssm:CreateOpsItem
	ssm:CreateOpsMetadata
	ssm:CreatePatchBaseline
	ssm:CreateResourceDataSync
	ssm:DeleteActivation
	ssm:DeleteAssociation
	ssm:DeleteCloudConnector
	ssm:DeleteDocument
	ssm:DeleteInventory
	ssm:DeleteMaintenanceWindow
	ssm:DeleteOpsItem
	ssm:DeleteOpsMetadata
	ssm:DeleteParameter

Prefixo do serviço	Ações do
	ssm:DeleteParameters
	ssm:DeletePatchBaseline
	ssm:DeleteResourceDataSync
	ssm:DeleteResourcePolicy
	ssm:DeregisterManagedInstance
	ssm:DeregisterPatchBaselineForPatchGroup
	ssm:DeregisterTargetFromMaintenanceWindow
	ssm:DeregisterTaskFromMaintenanceWindow
	ssm:DescribeActivations
	ssm:DescribeAssociation
	ssm:DescribeAssociationExecutionTargets
	ssm:DescribeAssociationExecutions
	ssm:DescribeAutomationExecutions
	ssm:DescribeAutomationStepExecutions
	ssm:DescribeAvailablePatches
	ssm:DescribeDocument
	ssm:DescribeDocumentParameters
	ssm:DescribeDocumentPermission
	ssm:DescribeEffectiveInstanceAssociations
	ssm:DescribeEffectivePatchesForPatchBaseline
	ssm:DescribeInstanceAssociationsStatus

Prefixo do serviço	Ações do
	ssm:DescribeInstanceInformation
	ssm:DescribeInstancePatchStates
	ssm:DescribeInstancePatchStatesForPatchGroup
	ssm:DescribeInstancePatches
	ssm:DescribeInstanceProperties
	ssm:DescribeInventoryDeletions
	ssm:DescribeMaintenanceWindowExecutionTaskInvocations
	ssm:DescribeMaintenanceWindowExecutionTasks
	ssm:DescribeMaintenanceWindowExecutions
	ssm:DescribeMaintenanceWindowSchedule
	ssm:DescribeMaintenanceWindowTargets
	ssm:DescribeMaintenanceWindowTasks
	ssm:DescribeMaintenanceWindows
	ssm:DescribeMaintenanceWindowsForTarget
	ssm:DescribeOpsItems
	ssm:DescribeParameters
	ssm:DescribePatchBaselines
	ssm:DescribePatchGroupState
	ssm:DescribePatchGroups
	ssm:DescribePatchProperties
	ssm:DescribeSessions

Prefixo do serviço	Ações do
	ssm:DisassociateOpsItemRelatedItem
	ssm:GetAccessToken
	ssm:GetAutomationExecution
	ssm:GetCalendarState
	ssm:GetCloudConnector
	ssm:GetCommandInvocation
	ssm:GetConnectionStatus
	ssm:GetDefaultPatchBaseline
	ssm:GetDeployablePatchSnapshotForInstance
	ssm:GetDocument
	ssm:GetExecutionPreview
	ssm:GetInventory
	ssm:GetInventorySchema
	ssm:GetMaintenanceWindow
	ssm:GetMaintenanceWindowExecution
	ssm:GetMaintenanceWindowExecutionTask
	ssm:GetMaintenanceWindowExecutionTaskInvocation
	ssm:GetMaintenanceWindowTask
	ssm:GetOpsItem
	ssm:GetOpsMetadata
	ssm:GetOpsSummary

Prefixo do serviço	Ações do
	ssm:GetParameter
	ssm:GetParameterHistory
	ssm:GetParameters
	ssm:GetParametersByPath
	ssm:GetPatchBaseline
	ssm:GetPatchBaselineForPatchGroup
	ssm:GetResourcePolicies
	ssm:GetServiceSetting
	ssm:LabelParameterVersion
	ssm:ListAssociationVersions
	ssm:ListAssociations
	ssm:ListCloudConnectors
	ssm:ListCommandInvocations
	ssm:ListCommands
	ssm:ListComplianceItems
	ssm:ListComplianceSummaries
	ssm:ListDocumentMetadataHistory
	ssm:ListDocumentVersions
	ssm:ListDocuments
	ssm:ListInstanceAssociations
	ssm:ListInventoryEntries

Prefixo do serviço	Ações do
	ssm:ListNodes
	ssm:ListNodesSummary
	ssm:ListOpsItemEvents
	ssm:ListOpsItemRelatedItems
	ssm:ListOpsMetadata
	ssm:ListResourceComplianceSummaries
	ssm:ListResourceDataSync
	ssm:ModifyDocumentPermission
	ssm:PutComplianceItems
	ssm:PutInventory
	ssm:PutParameter
	ssm:PutResourcePolicy
	ssm:RegisterDefaultPatchBaseline
	ssm:RegisterManagedInstance
	ssm:RegisterPatchBaselineForPatchGroup
	ssm:RegisterTargetWithMaintenanceWindow
	ssm:RegisterTaskWithMaintenanceWindow
	ssm:ResetServiceSetting
	ssm:ResumeSession
	ssm:SendAutomationSignal
	ssm:SendCommand

Prefixo do serviço	Ações do
	ssm:StartAssociationsOnce
	ssm:StartAutomationExecution
	ssm:StartChangeRequestExecution
	ssm:StartSession
	ssm:StopAutomationExecution
	ssm:TerminateSession
	ssm:UnlabelParameterVersion
	ssm:UpdateAssociation
	ssm:UpdateAssociationStatus
	ssm:UpdateCloudConnector
	ssm:UpdateDocument
	ssm:UpdateDocumentDefaultVersion
	ssm:UpdateDocumentMetadata
	ssm:UpdateInstanceInformation
	ssm:UpdateMaintenanceWindow
	ssm:UpdateMaintenanceWindowTarget
	ssm:UpdateMaintenanceWindowTask
	ssm:UpdateManagedInstanceRole
	ssm:UpdateOpsItem
	ssm:UpdateOpsMetadata
	ssm:UpdatePatchBaseline

Prefixo do serviço	Ações do
	ssm:UpdateResourceDataSync ssm:UpdateServiceSetting ssm:ValidateCloudConnector

Prefixo do serviço	Ações do
ssm-incidents	ssm-incidents:BatchGetIncidentFindings ssm-incidents:CreateReplicationSet ssm-incidents:CreateResponsePlan ssm-incidents:CreateTimelineEvent ssm-incidents:DeleteIncidentRecord ssm-incidents:DeleteReplicationSet ssm-incidents:DeleteResourcePolicy ssm-incidents:DeleteResponsePlan ssm-incidents:DeleteTimelineEvent ssm-incidents:GetIncidentRecord ssm-incidents:GetReplicationSet ssm-incidents:GetResourcePolicies ssm-incidents:GetResponsePlan ssm-incidents:GetTimelineEvent ssm-incidents:ListIncidentFindings ssm-incidents:ListIncidentRecords ssm-incidents:ListRelatedItems ssm-incidents:ListReplicationSets ssm-incidents:ListResponsePlans ssm-incidents:ListTimelineEvents ssm-incidents:PutResourcePolicy

Prefixo do serviço	Ações do
	ssm-incidents:StartIncident ssm-incidents:UpdateDeletionProtection ssm-incidents:UpdateIncidentRecord ssm-incidents:UpdateRelatedItems ssm-incidents:UpdateReplicationSet ssm-incidents:UpdateResponsePlan ssm-incidents:UpdateTimelineEvent

Prefixo do serviço	Ações do
ssm-sap	ssm-sap:BackupDatabase
	ssm-sap:DeleteResourcePermission
	ssm-sap:DeregisterApplication
	ssm-sap:GetApplication
	ssm-sap:GetComponent
	ssm-sap:GetConfigurationCheckOperation
	ssm-sap:GetDatabase
	ssm-sap:GetOperation
	ssm-sap:GetResourcePermission
	ssm-sap:ListApplications
	ssm-sap:ListComponents
	ssm-sap:ListConfigurationCheckDefinitions
	ssm-sap:ListConfigurationCheckOperations
	ssm-sap:ListDatabases
	ssm-sap:ListOperationEvents
	ssm-sap:ListOperations
	ssm-sap:ListSubCheckResults
	ssm-sap:ListSubCheckRuleResults
	ssm-sap:PutResourcePermission
	ssm-sap:RegisterApplication
	ssm-sap:RestoreDatabase

Prefixo do serviço	Ações do
	ssm-sap:StartApplication
	ssm-sap:StartApplicationRefresh
	ssm-sap:StartConfigurationChecks
	ssm-sap:StopApplication
	ssm-sap:UpdateApplicationSettings
	ssm-sap:UpdateHANABackupSettings

Prefixo do serviço	Ações do
estados	states:CreateActivity states:CreateStateMachine states:CreateStateMachineAlias states:DeleteActivity states:DeleteStateMachine states:DeleteStateMachineAlias states:DeleteStateMachineVersion states:DescribeActivity states:DescribeExecution states:DescribeMapRun states:DescribeStateMachine states:DescribeStateMachineAlias states:DescribeStateMachineForExecution states:GetExecutionHistory states:ListActivities states:ListExecutions states:ListMapRuns states:ListStateMachineAliases states:ListStateMachineVersions states:ListStateMachines states:SendTaskFailure

Prefixo do serviço	Ações do
	states:SendTaskHeartbeat states:SendTaskSuccess states:StartExecution states:StopExecution states:UpdateMapRun states:UpdateStateMachine states:UpdateStateMachineAlias states:ValidateStateMachineDefinition
sts	sts:AssumeRole sts:AssumeRoleWithSAML sts:AssumeRoleWithWebIdentity sts:DecodeAuthorizationMessage sts:GetAccessKeyInfo sts:GetCallerIdentity sts:GetFederationToken sts:GetSessionToken sts:GetWebIdentityToken

Prefixo do serviço	Ações do
swf	swf:DeleteActivityType swf:DeleteWorkflowType swf:DeprecateActivityType swf:DeprecateDomain swf:DeprecateWorkflowType swf:DescribeActivityType swf:DescribeDomain swf:DescribeWorkflowType swf:ListActivityTypes swf:ListDomains swf:ListWorkflowTypes swf:RegisterActivityType swf:RegisterDomain swf:RegisterWorkflowType swf:UndeprecateActivityType swf:UndeprecateDomain swf:UndeprecateWorkflowType

Prefixo do serviço	Ações do
synthetics	synthetics:AssociateResource
	synthetics:CreateCanary
	synthetics:CreateGroup
	synthetics:DeleteCanary
	synthetics:DeleteGroup
	synthetics:DescribeCanaries
	synthetics:DescribeCanariesLastRun
	synthetics:DescribeRuntimeVersions
	synthetics:DisassociateResource
	synthetics:GetCanary
	synthetics:GetCanaryRuns
	synthetics:GetGroup
	synthetics:ListAssociatedGroups
	synthetics:ListGroupResources
	synthetics:ListGroups
	synthetics:StartCanary
	synthetics:StartCanaryDryRun
	synthetics:StopCanary
	synthetics:UpdateCanary

Prefixo do serviço	Ações do
tag	tag:DescribeReportCreation tag:GetComplianceSummary tag:GetResources tag:StartReportCreation

Prefixo do serviço	Ações do
textract	textract:AnalyzeDocument
	textract:AnalyzeExpense
	textract:AnalyzeID
	textract:CreateAdapter
	textract:CreateAdapterVersion
	textract:DeleteAdapter
	textract:DeleteAdapterVersion
	textract:DetectDocumentText
	textract:GetAdapter
	textract:GetAdapterVersion
	textract:GetDocumentAnalysis
	textract:GetDocumentTextDetection
	textract:GetExpenseAnalysis
	textract:GetLendingAnalysis
	textract:GetLendingAnalysisSummary
	textract:ListAdapterVersions
	textract:ListAdapters
	textract:StartDocumentAnalysis
	textract:StartDocumentTextDetection
	textract:StartExpenseAnalysis
	textract:StartLendingAnalysis

Prefixo do serviço	Ações do
	textract:UpdateAdapter

Prefixo do serviço	Ações do
timestream	timestream:CancelQuery timestream:CreateDatabase timestream:CreateScheduledQuery timestream:CreateTable timestream:DeleteDatabase timestream:DeleteScheduledQuery timestream:DeleteTable timestream:DescribeAccountSettings timestream:DescribeDatabase timestream:DescribeScheduledQuery timestream:DescribeTable timestream:ExecuteScheduledQuery timestream:ListBatchLoadTasks timestream:ListDatabases timestream:ListScheduledQueries timestream:ListTables timestream:PrepareQuery timestream:UpdateAccountSettings timestream:UpdateDatabase timestream:UpdateScheduledQuery timestream:UpdateTable

Prefixo do serviço	Ações do
tnb	tnb:CancelSolNetworkOperation
	tnb:CreateSolFunctionPackage
	tnb:CreateSolNetworkInstance
	tnb:CreateSolNetworkPackage
	tnb:DeleteSolFunctionPackage
	tnb:DeleteSolNetworkInstance
	tnb:DeleteSolNetworkPackage
	tnb:GetSolFunctionInstance
	tnb:GetSolFunctionPackage
	tnb:GetSolFunctionPackageContent
	tnb:GetSolFunctionPackageDescriptor
	tnb:GetSolNetworkInstance
	tnb:GetSolNetworkOperation
	tnb:GetSolNetworkPackage
	tnb:GetSolNetworkPackageContent
	tnb:GetSolNetworkPackageDescriptor
	tnb:InstantiateSolNetworkInstance
	tnb:ListSolFunctionInstances
	tnb:ListSolFunctionPackages
	tnb:ListSolNetworkInstances
	tnb:ListSolNetworkOperations

Prefixo do serviço	Ações do
	tnb:ListSolNetworkPackages
	tnb:PutSolFunctionPackageContent
	tnb:PutSolNetworkPackageContent
	tnb:TerminateSolNetworkInstance
	tnb:UpdateSolFunctionPackage
	tnb:UpdateSolNetworkInstance
	tnb:UpdateSolNetworkPackage
	tnb:ValidateSolFunctionPackageContent
	tnb:ValidateSolNetworkPackageContent

Prefixo do serviço	Ações do
transcribe	transcribe:CreateCallAnalyticsCategory transcribe:CreateLanguageModel transcribe:CreateMedicalVocabulary transcribe:CreateVocabulary transcribe:CreateVocabularyFilter transcribe:DeleteCallAnalyticsCategory transcribe:DeleteCallAnalyticsJob transcribe:DeleteLanguageModel transcribe:DeleteMedicalScribeJob transcribe:DeleteMedicalTranscriptionJob transcribe:DeleteMedicalVocabulary transcribe:DeleteTranscriptionJob transcribe:DeleteVocabulary transcribe:DeleteVocabularyFilter transcribe:DescribeLanguageModel transcribe:GetCallAnalyticsCategory transcribe:GetCallAnalyticsJob transcribe:GetMedicalScribeJob transcribe:GetMedicalTranscriptionJob transcribe:GetMedicalVocabulary transcribe:GetTranscriptionJob

Prefixo do serviço	Ações do
	transcribe:GetVocabulary transcribe:GetVocabularyFilter transcribe:ListCallAnalyticsCategories transcribe:ListCallAnalyticsJobs transcribe:ListLanguageModels transcribe:ListMedicalScribeJobs transcribe:ListMedicalTranscriptionJobs transcribe:ListMedicalVocabularies transcribe:ListTranscriptionJobs transcribe:ListVocabularies transcribe:ListVocabularyFilters transcribe:StartCallAnalyticsJob transcribe:StartCallAnalyticsStreamTranscription transcribe:StartCallAnalyticsStreamTranscriptionWebSocket transcribe:StartMedicalScribeJob transcribe:StartMedicalStreamTranscription transcribe:StartMedicalStreamTranscriptionWebSocket transcribe:StartMedicalTranscriptionJob transcribe:StartStreamTranscription transcribe:StartStreamTranscriptionWebSocket transcribe:StartTranscriptionJob

Prefixo do serviço	Ações do
	transcribe:UpdateCallAnalyticsCategory transcribe:UpdateMedicalVocabulary transcribe:UpdateVocabulary transcribe:UpdateVocabularyFilter

Prefixo do serviço	Ações do
transferência	transfer:CreateAccess
	transfer:CreateAgreement
	transfer:CreateConnector
	transfer:CreateProfile
	transfer:CreateServer
	transfer:CreateUser
	transfer:CreateWebApp
	transfer:CreateWorkflow
	transfer:DeleteAccess
	transfer:DeleteAgreement
	transfer:DeleteCertificate
	transfer:DeleteConnector
	transfer:DeleteHostKey
	transfer:DeleteProfile
	transfer:DeleteServer
	transfer:DeleteSshPublicKey
	transfer:DeleteUser
	transfer:DeleteWebApp
	transfer:DeleteWebAppCustomization
	transfer:DeleteWorkflow
	transfer:DescribeAccess

Prefixo do serviço	Ações do
	transfer:DescribeAgreement
	transfer:DescribeCertificate
	transfer:DescribeConnector
	transfer:DescribeExecution
	transfer:DescribeHostKey
	transfer:DescribeProfile
	transfer:DescribeSecurityPolicy
	transfer:DescribeServer
	transfer:DescribeUser
	transfer:DescribeWebApp
	transfer:DescribeWebAppCustomization
	transfer:DescribeWorkflow
	transfer:ImportCertificate
	transfer:ImportHostKey
	transfer:ImportSshPublicKey
	transfer:ListAccesses
	transfer:ListCertificates
	transfer:ListConnectors
	transfer:ListExecutions
	transfer:ListFileTransferResults
	transfer:ListHostKeys

Prefixo do serviço	Ações do
	transfer:ListProfiles
	transfer:ListSecurityPolicies
	transfer:ListServers
	transfer:ListUsers
	transfer:ListWebApps
	transfer:ListWorkflows
	transfer:SendWorkflowStepState
	transfer:StartDirectoryListing
	transfer:StartFileTransfer
	transfer:StartRemoteDelete
	transfer:StartRemoteMove
	transfer:StartServer
	transfer:StopServer
	transfer:TestConnection
	transfer:TestIdentityProvider
	transfer:UpdateAccess
	transfer:UpdateAgreement
	transfer:UpdateCertificate
	transfer:UpdateConnector
	transfer:UpdateHostKey
	transfer:UpdateProfile

Prefixo do serviço	Ações do
	transfer:UpdateServer transfer:UpdateUser transfer:UpdateWebApp transfer:UpdateWebAppCustomization
translate	translate:CreateParallelData translate:DeleteParallelData translate:DeleteTerminology translate:DescribeTextTranslationJob translate:GetParallelData translate:GetTerminology translate:ImportTerminology translate:ListLanguages translate:ListParallelData translate:ListTerminologies translate:ListTextTranslationJobs translate:StartTextTranslationJob translate:StopTextTranslationJob translate:TranslateDocument translate:TranslateText translate:UpdateParallelData

Prefixo do serviço	Ações do
voiceid	voiceid:AssociateFraudster
	voiceid:CreateDomain
	voiceid:CreateWatchlist
	voiceid:DeleteDomain
	voiceid:DeleteFraudster
	voiceid:DeleteSpeaker
	voiceid:DeleteWatchlist
	voiceid:DescribeDomain
	voiceid:DescribeFraudster
	voiceid:DescribeFraudsterRegistrationJob
	voiceid:DescribeSpeaker
	voiceid:DescribeSpeakerEnrollmentJob
	voiceid:DescribeWatchlist
	voiceid:DisassociateFraudster
	voiceid:EvaluateSession
	voiceid:ListDomains
	voiceid:ListFraudsterRegistrationJobs
	voiceid:ListFraudsters
	voiceid:ListSpeakerEnrollmentJobs
	voiceid:ListSpeakers
	voiceid:ListWatchlists

Prefixo do serviço	Ações do
	voiceid:OptOutSpeaker voiceid:StartFraudsterRegistrationJob voiceid:StartSpeakerEnrollmentJob voiceid:UpdateDomain voiceid:UpdateWatchlist

Prefixo do serviço	Ações do
vpc-lattice	vpc-lattice:CreateAccessLogSubscription vpc-lattice:CreateListener vpc-lattice:CreateResourceConfiguration vpc-lattice:CreateResourceGateway vpc-lattice:CreateRule vpc-lattice:CreateService vpc-lattice:CreateServiceNetwork vpc-lattice:CreateServiceNetworkResourceAssociation vpc-lattice:CreateServiceNetworkServiceAssociation vpc-lattice:CreateServiceNetworkVpcAssociation vpc-lattice:CreateTargetGroup vpc-lattice:DeleteAccessLogSubscription vpc-lattice:DeleteAuthPolicy vpc-lattice:DeleteDomainVerification vpc-lattice:DeleteListener vpc-lattice:DeleteResourceConfiguration vpc-lattice:DeleteResourceEndpointAssociation vpc-lattice:DeleteResourceGateway vpc-lattice:DeleteResourcePolicy vpc-lattice:DeleteRule vpc-lattice:DeleteService

Prefixo do serviço	Ações do
	vpc-lattice:DeleteServiceNetwork vpc-lattice:DeleteServiceNetworkResourceAssociation vpc-lattice:DeleteServiceNetworkServiceAssociation vpc-lattice:DeleteServiceNetworkVpcAssociation vpc-lattice:DeleteTargetGroup vpc-lattice:DeregisterTargets vpc-lattice:GetAccessLogSubscription vpc-lattice:GetAuthPolicy vpc-lattice:GetDomainVerification vpc-lattice:GetListener vpc-lattice:GetResourceConfiguration vpc-lattice:GetResourceGateway vpc-lattice:GetResourcePolicy vpc-lattice:GetRule vpc-lattice:GetService vpc-lattice:GetServiceNetwork vpc-lattice:GetServiceNetworkResourceAssociation vpc-lattice:GetServiceNetworkServiceAssociation vpc-lattice:GetServiceNetworkVpcAssociation vpc-lattice:GetTargetGroup vpc-lattice:ListAccessLogSubscriptions

Prefixo do serviço	Ações do
	vpc-lattice:ListDomainVerifications vpc-lattice:ListListeners vpc-lattice:ListResourceConfigurations vpc-lattice:ListResourceEndpointAssociations vpc-lattice:ListResourceGateways vpc-lattice:ListRules vpc-lattice:ListServiceNetworkResourceAssociations vpc-lattice:ListServiceNetworkServiceAssociations vpc-lattice:ListServiceNetworkVpcAssociations vpc-lattice:ListServiceNetworkVpcEndpointAssociations vpc-lattice:ListServiceNetworks vpc-lattice:ListServices vpc-lattice:ListTargetGroups vpc-lattice:ListTargets vpc-lattice:PutAuthPolicy vpc-lattice:PutResourcePolicy vpc-lattice:RegisterTargets vpc-lattice:StartDomainVerification vpc-lattice:UpdateAccessLogSubscription vpc-lattice:UpdateListener vpc-lattice:UpdateResourceConfiguration

Prefixo do serviço	Ações do
	vpc-lattice:UpdateResourceGateway vpc-lattice:UpdateRule vpc-lattice:UpdateService vpc-lattice:UpdateServiceNetwork vpc-lattice:UpdateServiceNetworkVpcAssociation vpc-lattice:UpdateTargetGroup

Prefixo do serviço	Ações do
wafv2	wafv2:AssociateWebACL
	wafv2:CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2:CreateRegexPatternSet
	wafv2:CreateRuleGroup
	wafv2:CreateWebACL
	wafv2:DeleteAPIKey
	wafv2:DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2:DeleteLoggingConfiguration
	wafv2:DeletePermissionPolicy
	wafv2:DeleteRegexPatternSet
	wafv2:DeleteRuleGroup
	wafv2:DeleteWebACL
	wafv2:DescribeAllManagedProducts
	wafv2:DescribeManagedProductsByVendor
	wafv2:DescribeManagedRuleGroup
	wafv2:DisassociateWebACL
	wafv2:GenerateMobileSdkReleaseUrl
	wafv2:GetDecryptedAPIKey

Prefixo do serviço	Ações do
	wafv2:GetIPSet
	wafv2:GetLoggingConfiguration
	wafv2:GetManagedRuleSet
	wafv2:GetMobileSdkRelease
	wafv2:GetRateBasedStatementManagedKeys
	wafv2:GetRegexPatternSet
	wafv2:GetRevenueStatistics
	wafv2:GetRevenueStatisticsSummary
	wafv2:GetRevenueStatisticsTimeSeries
	wafv2:GetRuleGroup
	wafv2:GetSampledRequests
	wafv2:GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2:ListAvailableManagedRuleGroupVersions
	wafv2:ListAvailableManagedRuleGroups
	wafv2:ListIPSets
	wafv2:ListLoggingConfigurations
	wafv2:ListManagedRuleSets
	wafv2:ListMobileSdkReleases
	wafv2:ListRegexPatternSets
	wafv2:ListResourcesForWebACL

Prefixo do serviço	Ações do
	wafv2:ListRuleGroups
	wafv2:ListSettlementRecords
	wafv2:ListWebACLs
	wafv2:PutLoggingConfiguration
	wafv2:PutManagedRuleSetVersions
	wafv2:UpdateIPSet
	wafv2:UpdateManagedRuleSetVersionExpiryDate
	wafv2:UpdateRegexPatternSet
	wafv2:UpdateRuleGroup
	wafv2:UpdateWebACL

Prefixo do serviço	Ações do
wellarchitected	wellarchitected:AssociateLenses wellarchitected:AssociateProfiles wellarchitected:CreateLensShare wellarchitected:CreateLensVersion wellarchitected:CreateMilestone wellarchitected:CreateProfile wellarchitected:CreateProfileShare wellarchitected:CreateReviewTemplate wellarchitected:CreateWorkload wellarchitected:CreateWorkloadShare wellarchitected:DeleteLens wellarchitected:DeleteLensShare wellarchitected:DeleteProfile wellarchitected:DeleteProfileShare wellarchitected:DeleteReviewTemplate wellarchitected:DeleteTemplateShare wellarchitected:DeleteWorkload wellarchitected:DeleteWorkloadShare wellarchitected:DisassociateLenses wellarchitected:DisassociateProfiles wellarchitected:ExportLens

Prefixo do serviço	Ações do
	wellarchitected:GetAnswer wellarchitected:GetConsolidatedReport wellarchitected:GetGlobalSettings wellarchitected:GetLens wellarchitected:GetLensReview wellarchitected:GetLensReviewReport wellarchitected:GetLensVersionDifference wellarchitected:GetMilestone wellarchitected:GetProfile wellarchitected:GetProfileTemplate wellarchitected:GetReviewTemplate wellarchitected:GetReviewTemplateAnswer wellarchitected:GetReviewTemplateLensReview wellarchitected:GetWorkload wellarchitected:ImportLens wellarchitected:ListAnswers wellarchitected:ListCheckDetails wellarchitected:ListCheckSummaries wellarchitected:ListLensReviewImprovements wellarchitected:ListLensReviews wellarchitected:ListLensShares

Prefixo do serviço	Ações do
	wellarchitected:ListLenses wellarchitected:ListMilestones wellarchitected:ListNotifications wellarchitected:ListProfileNotifications wellarchitected:ListProfileShares wellarchitected:ListProfiles wellarchitected:ListReviewTemplateAnswers wellarchitected:ListReviewTemplates wellarchitected:ListShareInvitations wellarchitected:ListTemplateShares wellarchitected:ListWorkloadShares wellarchitected:ListWorkloads wellarchitected:UpdateAnswer wellarchitected:UpdateGlobalSettings wellarchitected:UpdateIntegration wellarchitected:UpdateLensReview wellarchitected:UpdateProfile wellarchitected:UpdateReviewTemplate wellarchitected:UpdateReviewTemplateLensReview wellarchitected:UpdateShareInvitation wellarchitected:UpdateWorkload

Prefixo do serviço	Ações do
	wellarchitected:UpdateWorkloadShare wellarchitected:UpgradeLensReview wellarchitected:UpgradeProfileVersion wellarchitected:UpgradeReviewTemplateLensReview

Prefixo do serviço	Ações do
wisdom	wisdom:CreateAssistant
	wisdom:CreateAssistantAssociation
	wisdom:CreateContent
	wisdom:CreateKnowledgeBase
	wisdom:CreateQuickResponse
	wisdom:CreateSession
	wisdom:DeleteAssistant
	wisdom:DeleteAssistantAssociation
	wisdom:DeleteContent
	wisdom:DeleteImportJob
	wisdom:DeleteKnowledgeBase
	wisdom:DeleteQuickResponse
	wisdom:GetAssistant
	wisdom:GetAssistantAssociation
	wisdom:GetContent
	wisdom:GetContentAssociation
	wisdom:GetContentSummary
	wisdom:GetImportJob
	wisdom:GetKnowledgeBase
	wisdom:GetRecommendations
	wisdom:GetSession

Prefixo do serviço	Ações do
	wisdom:ListAssistantAssociations
	wisdom:ListAssistants
	wisdom:ListContentAssociations
	wisdom:ListContents
	wisdom:ListImportJobs
	wisdom:ListKnowledgeBases
	wisdom:ListQuickResponses
	wisdom:NotifyRecommendationsReceived
	wisdom:QueryAssistant
	wisdom:RemoveKnowledgeBaseTemplateUri
	wisdom:SearchContent
	wisdom:SearchQuickResponses
	wisdom:SearchSessions
	wisdom:StartContentUpload
	wisdom:StartImportJob
	wisdom:UpdateContent
	wisdom:UpdateKnowledgeBaseTemplateUri
	wisdom:UpdateQuickResponse
	wisdom:UpdateSession

Prefixo do serviço	Ações do
worklink	worklink:AssociateDomain worklink:AssociateWebsiteAuthorizationProvider worklink:AssociateWebsiteCertificateAuthority worklink:CreateFleet worklink:DeleteFleet worklink:DescribeAuditStreamConfiguration worklink:DescribeCompanyNetworkConfiguration worklink:DescribeDevice worklink:DescribeDevicePolicyConfiguration worklink:DescribeDomain worklink:DescribeFleetMetadata worklink:DescribeIdentityProviderConfiguration worklink:DescribeWebsiteCertificateAuthority worklink:DisassociateDomain worklink:DisassociateWebsiteAuthorizationProvider worklink:DisassociateWebsiteCertificateAuthority worklink:ListDevices worklink:ListDomains worklink:ListFleets worklink:ListWebsiteAuthorizationProviders worklink:ListWebsiteCertificateAuthorities

Prefixo do serviço	Ações do
	worklink:RestoreDomainAccess worklink:RevokeDomainAccess worklink:SignOutUser worklink:UpdateAuditStreamConfiguration worklink:UpdateCompanyNetworkConfiguration worklink:UpdateDevicePolicyConfiguration worklink:UpdateDomainMetadata worklink:UpdateFleetMetadata worklink:UpdateIdentityProviderConfiguration

Prefixo do serviço	Ações do
espaços de trabalho	workspaces:AcceptAccountLinkInvitation
	workspaces:AssociateConnectionAlias
	workspaces:AssociateIpGroups
	workspaces:AssociateWorkspaceApplication
	workspaces:CopyWorkspaceImage
	workspaces:CreateAccountLinkInvitation
	workspaces:CreateConnectClientAddIn
	workspaces:CreateConnectionAlias
	workspaces:CreateIpGroup
	workspaces:CreateStandbyWorkspaces
	workspaces:CreateUpdatedWorkspaceImage
	workspaces:CreateWorkspaceBundle
	workspaces:CreateWorkspaceImage
	workspaces:CreateWorkspaces
	workspaces:CreateWorkspacesPool
	workspaces:DeleteAccountLinkInvitation
	workspaces:DeleteClientBranding
	workspaces:DeleteConnectClientAddIn
	workspaces:DeleteConnectionAlias
	workspaces:DeleteIpGroup
	workspaces:DeleteWorkspaceBundle

Prefixo do serviço	Ações do
	workspaces:DeleteWorkspaceImage
	workspaces:DeployWorkspaceApplications
	workspaces:DeregisterWorkspaceDirectory
	workspaces:DescribeAccount
	workspaces:DescribeAccountModifications
	workspaces:DescribeApplicationAssociations
	workspaces:DescribeApplications
	workspaces:DescribeBundleAssociations
	workspaces:DescribeClientBranding
	workspaces:DescribeClientProperties
	workspaces:DescribeConnectClientAddIns
	workspaces:DescribeConnectionAliasPermissions
	workspaces:DescribeConnectionAliases
	workspaces:DescribeCustomWorkspaceImageImport
	workspaces:DescribeImageAssociations
	workspaces:DescribeIpGroups
	workspaces:DescribeWorkspaceAssociations
	workspaces:DescribeWorkspaceBundles
	workspaces:DescribeWorkspaceDirectories
	workspaces:DescribeWorkspaceImagePermissions
	workspaces:DescribeWorkspaceSnapshots

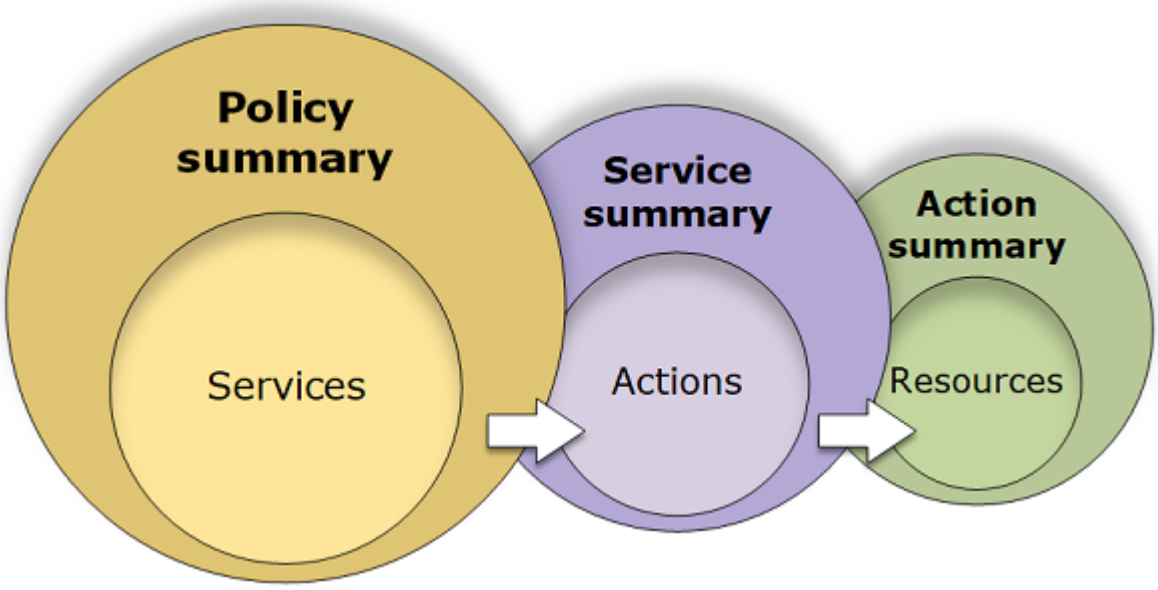
Prefixo do serviço	Ações do
	workspaces:DescribeWorkspaces
	workspaces:DescribeWorkspacesConnectionStatus
	workspaces:DescribeWorkspacesPoolSessions
	workspaces:DescribeWorkspacesPools
	workspaces:DisassociateConnectionAlias
	workspaces:DisassociateIpGroups
	workspaces:DisassociateWorkspaceApplication
	workspaces:GetAccountLink
	workspaces:ImportClientBranding
	workspaces:ImportWorkspaceImage
	workspaces:ListAccountLinks
	workspaces:ListAvailableManagementCidrRanges
	workspaces:MigrateWorkspace
	workspaces:ModifyAccount
	workspaces:ModifyCertificateBasedAuthProperties
	workspaces:ModifyClientProperties
	workspaces:ModifyEndpointEncryptionMode
	workspaces:ModifySamlProperties
	workspaces:ModifySelfservicePermissions
	workspaces:ModifyStreamingProperties
	workspaces:ModifyWorkspaceAccessProperties

Prefixo do serviço	Ações do
	workspaces:ModifyWorkspaceCreationProperties
	workspaces:ModifyWorkspaceProperties
	workspaces:ModifyWorkspaceState
	workspaces:RebootWorkspaces
	workspaces:RebuildWorkspaces
	workspaces:RegisterWorkspaceDirectory
	workspaces:RejectAccountLinkInvitation
	workspaces:RestoreWorkspace
	workspaces:StartWorkspaces
	workspaces:StartWorkspacesPool
	workspaces:StopWorkspaces
	workspaces:StopWorkspacesPool
	workspaces:TerminateWorkspaces
	workspaces:TerminateWorkspacesPool
	workspaces:TerminateWorkspacesPoolSession
	workspaces:UpdateConnectClientAddIn
	workspaces:UpdateConnectionAliasPermission
	workspaces:UpdateWorkspaceBundle
	workspaces:UpdateWorkspaceImagePermission
	workspaces:UpdateWorkspacesPool

Prefixo do serviço	Ações do
xray	xray:CreateGroup xray:CreateSamplingRule xray:DeleteGroup xray:DeleteResourcePolicy xray:DeleteSamplingRule xray:GetEncryptionConfig xray:GetGroup xray:GetGroups xray:GetInsight xray:GetInsightEvents xray:GetInsightImpactGraph xray:GetInsightSummaries xray:GetSamplingRules xray:ListResourcePolicies xray:PutEncryptionConfig xray:PutResourcePolicy xray:UpdateGroup xray:UpdateSamplingRule

Resumos de políticas

O console do IAM inclui tabelas do resumo de políticas que descrevem o nível de acesso, os recursos e as condições permitidas ou negadas para cada serviço em uma política. As políticas são resumidas em três tabelas: o resumo de políticas, o resumo de serviços e o resumo de ações. A tabela resumo da política inclui uma lista de serviços. Escolha um serviço para ver o resumo do serviço. Esta tabela de resumo inclui uma lista das ações e permissões associadas para o serviço escolhido. Você pode escolher uma ação dessa tabela para visualizar o resumo da ação. Esta tabela inclui uma lista de recursos e condições para a ação escolhida.



Visualize os resumos das políticas nas páginas Usuários ou Funções para todas as políticas (gerenciadas e em linha) anexadas a esse usuário. Visualize os resumos na página Políticas para todas as políticas gerenciadas. As políticas gerenciadas incluem políticas gerenciadas pela AWS, políticas de função de cargo gerenciadas pela AWS e políticas gerenciadas pelo cliente. Você pode exibir resumos dessas políticas na página Policies (Políticas), independentemente de estarem anexadas ou não a um usuário ou outra identidade do IAM.

Você pode usar as informações nos resumos de política para compreender o que é permitido ou negado por sua política. Os resumos de política podem ajudá-lo a solucionar problemas e corrigir políticas que não estão fornecendo as permissões esperadas.

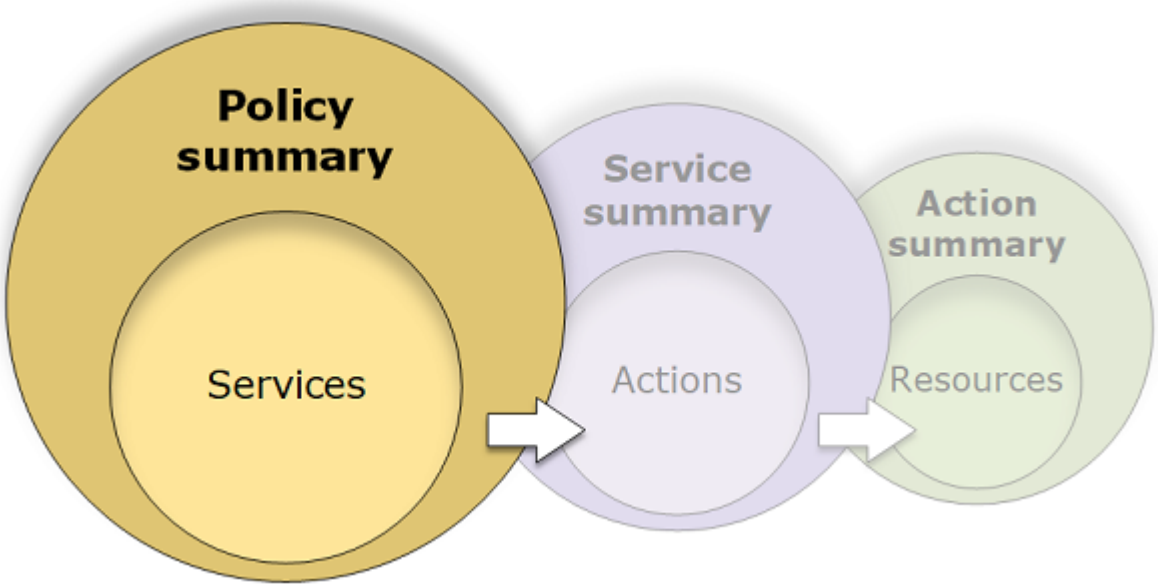
Para obter instruções passo a passo sobre a visualização de resumos, consulte Visualizar resumos de políticas, Visualizar resumos de serviços e Visualizar resumos de ações.

Tópicos

- Resumo da política (lista de serviços)
- Níveis de acesso em resumos de política
- Resumo do serviço (lista de ações)
- Resumo da ação (lista de recursos)
- Exemplos de resumos de políticas

Resumo da política (lista de serviços)

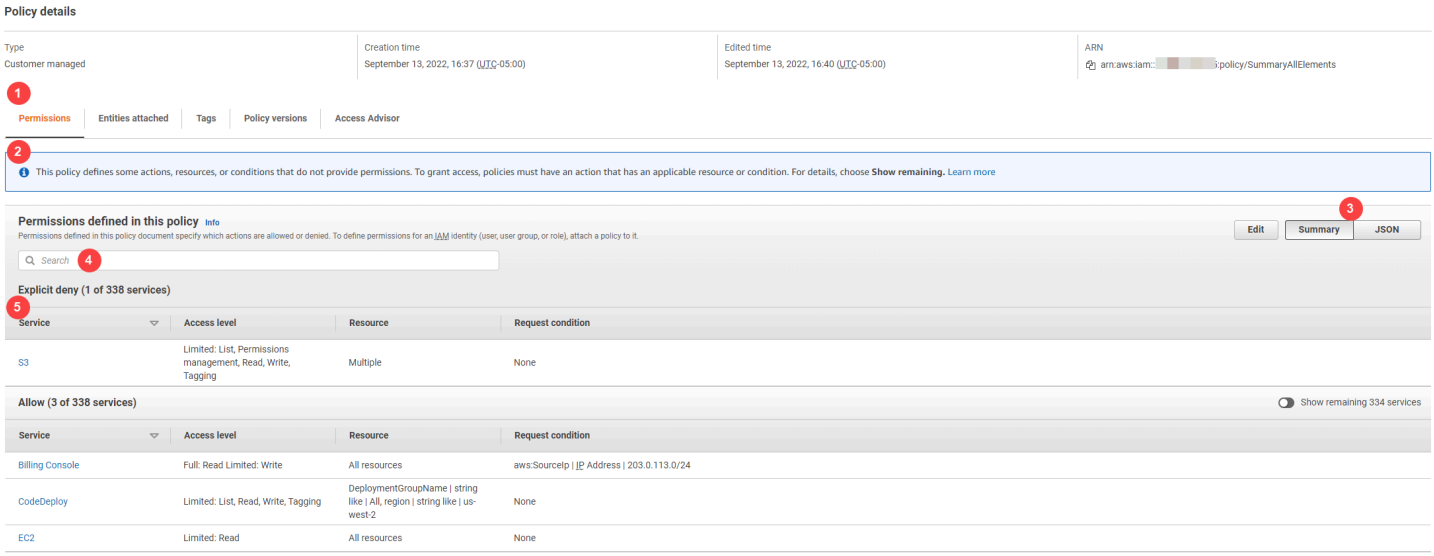
As políticas são resumidas em três tabelas: o resumo de políticas, o resumo de serviços e o resumo de ações. A tabela de resumo da política inclui uma lista de serviços e resumos das permissões que são definidas pela política escolhida.



A tabela de resumo da política é agrupada em um ou mais seções de Serviços não categorizados, Negação explícita e Permissão. Se a política incluir um serviço que o IAM não reconheça, o serviço será incluído na seção Uncategorized services (Serviços não categorizados) da tabela. Se o IAM reconhecer o serviço, ele será incluído nas seções Explicit deny (Negação explícita) ou Allow (Permitir) da tabela, dependendo do efeito da política (`Deny` ou `Allow`).

Noções básicas sobre os elementos de um resumo de política

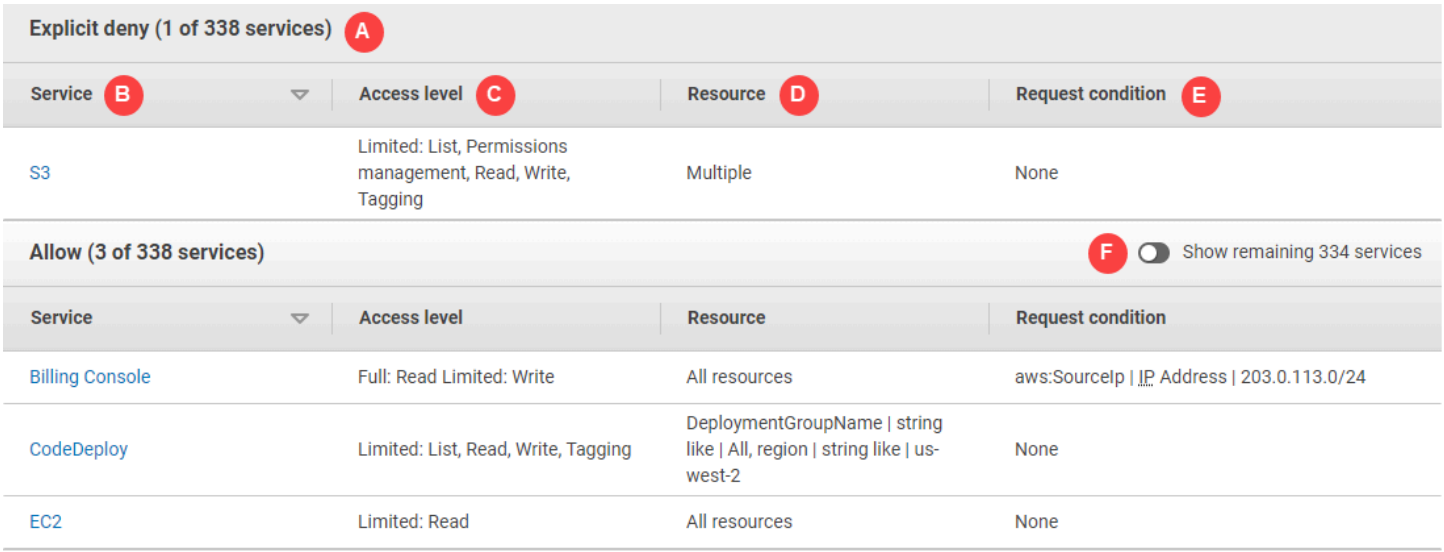
No exemplo de página de detalhes de uma política a seguir, a política SummaryAllElements é uma política gerenciada (pelo cliente) que está anexada diretamente ao usuário. Essa política é expandida para mostrar seu resumo.



Na imagem anterior, o resumo da política pode ser visto na página Políticas:

1. A guia Permissões inclui as permissões definidas na política.
2. Se a política não conceder permissões para todas as ações, recursos e condições definidos na política, um banner de aviso ou erro aparecerá na parte superior da página. O resumo de política inclui detalhes sobre o problema. Para saber como os resumos de política ajudam a entender e solucionar problemas das permissões concedidas por sua política, consulte the section called “Minha política não concede as permissões esperadas”.
3. Use os botões Resumo e JSON para alternar entre o resumo da política e o documento da política JSON.
4. Use a caixa Pesquisar para reduzir a lista de serviços e localizar um serviço específico.
5. A visualização expandida mostra detalhes adicionais da política SummaryAllElements.

A imagem de tabela do resumo da política seguir mostra a política SummaryAllElements expandida na página de detalhes da política.



Explicit deny (1 of 338 services) A

Service B	Access level C	Resource D	Request condition E
S3	Limited: List, Permissions management, Read, Write, Tagging	Multiple	None

Allow (3 of 338 services) F Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeDeploy	Limited: List, Read, Write, Tagging	DeploymentGroupName | string like | All, region | string like | us-west-2	None
EC2	Limited: Read	All resources	None

Na imagem anterior, o resumo da política pode ser visto na página Políticas:

A. Para os serviços reconhecidos pelo IAM, ele organiza os serviços de acordo com a natureza da política, ou seja, se ela permite ou nega explicitamente o uso do serviço. Neste exemplo, a política inclui uma instrução `Deny` para o serviço Amazon S3 e instruções `Allow` para os serviços Faturamento, CodeDeploy e Amazon EC2.

B. Service (Serviço): esta coluna lista os serviços que estão definidos na política e fornece detalhes de cada serviço. Cada nome do serviço na tabela de resumo da política é um link para a tabela resumo do serviço, que é explicada em Resumo do serviço (lista de ações). Neste exemplo, as permissões estão definidas para os serviços Amazon S3, Faturamento, CodeDeploy e Amazon EC2.

C. Nível de acesso: essa coluna informa se as ações em cada nível de acesso (`List`, `Read`, `Write`, `Permission Management` e `Tagging`) têm as permissões `Full` ou `Limited` definidas na política. Para obter mais detalhes e exemplos do resumo de nível de acesso, consulte Níveis de acesso em resumos de política.

- Full access (Acesso total): esta opção indica que o serviço tem acesso a todas as ações em todos os quatro níveis de acesso disponíveis para o serviço.
- Se a entrada não incluir Acesso total, o serviço terá acesso a algumas, mas não a todas as ações para o serviço. O acesso é, então, definido seguindo as descrições de cada uma das classificações de nível de acesso (`List`, `Read`, `Write`, `Permission Management` e `Tagging`):

  Total: a política fornece acesso a todas as ações em cada classificação de nível de acesso listada. Neste exemplo, a política fornece acesso a todas as ações `Read` de faturamento.

Limitado: a política fornece acesso a uma ou mais, mas não todas as ações em cada classificação de nível de acesso listada. Neste exemplo, a política fornece acesso a algumas ações `Write` de faturamento.

D. Resource (Recurso): esta coluna mostra os recursos que a política especifica para cada serviço.

- Multiple (Múltiplo): a política inclui mais de um, mas não todos os recursos do serviço. Neste exemplo, o acesso é negado explicitamente para mais de um recurso do Amazon S3.
- Todos os recursos: a política é definida para todos os recursos do serviço. Neste exemplo, a política permite que as ações listadas sejam executadas em todos os recursos de faturamento.
- Texto do recurso: a política inclui um recurso do serviço. Neste exemplo, as ações listadas só são permitidas no recurso `DeploymentGroupName` de CodeDeploy. Dependendo das informações que o serviço fornecer ao IAM, você poderá ver um ARN ou o tipo de recurso definido.

> **Note**
>
> Essa coluna pode incluir um recurso de um serviço diferente. Se a declaração de política que inclui o recurso não incluir as duas ações e recursos do mesmo serviço, a política incluirá recursos incompatíveis. O IAM não avisa sobre recursos incompatíveis quando você cria uma política, ou quando visualiza uma política no resumo da política. Se essa coluna incluir um recurso incompatível, será necessário analisar sua política para busca de erros. Para compreender melhor suas políticas, sempre teste-as com o simulador de políticas.

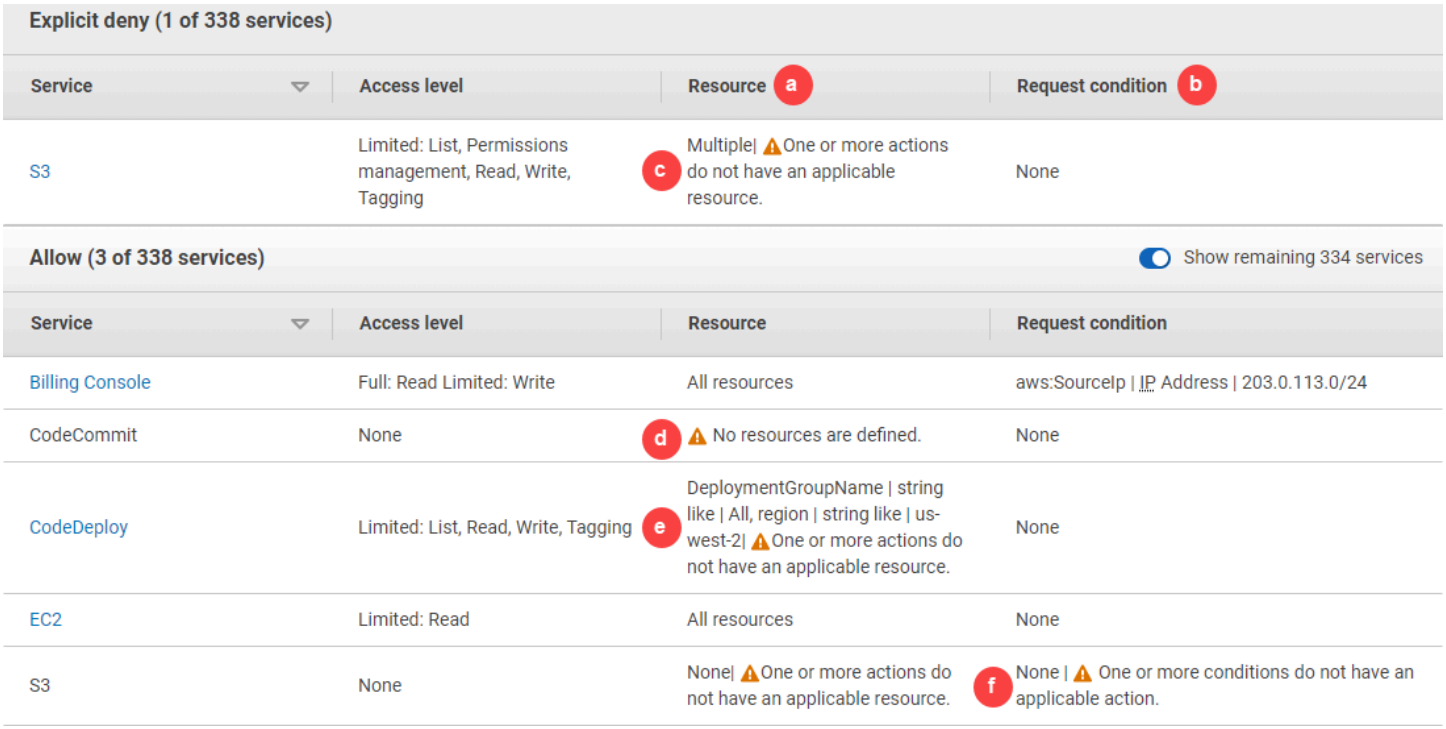
E. Request condition (Condição de solicitação): esta coluna indica se os serviços ou ações associadas ao recurso estão sujeitos a condições.

- None (Nenhum): a política não inclui condições para o serviço. Neste exemplo, nenhuma condição é aplicada às ações negadas no serviço Amazon S3.
- Texto da condição: a política inclui uma condição para o serviço. Neste exemplo, as ações de Faturamento listadas só serão permitidas se o endereço IP da fonte corresponder a `203.0.113.0/24`.
- Multiple (Vários): a política inclui mais de uma condição para o serviço. Para visualizar cada uma das várias condições da política, escolha JSON para visualizar o documento da política.

F. Mostrar serviços restantes: alterne esse botão para expandir a tabela e incluir os serviços que não são definidos pela política. Esses serviços são negados implicitamente (ou negados por padrão) dentro dessa política. No entanto, uma declaração em outra política ainda pode permitir

ou negar explicitamente usando o serviço. O resumo da política resume as permissões de uma única política. Para saber como o serviço da AWS decide se uma determinada solicitação deve ser permitida ou negada, consulte Lógica da avaliação de política.

Quando uma política ou um elemento dentro da política não concede permissões, o IAM fornece mais avisos e informações no resumo da política. A tabela de resumo da política a seguir mostra os serviços Mostrar serviços restantes expandidos na página de detalhes da política SummaryAllElements com os possíveis avisos.



Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

Na imagem anterior, você pode ver todos os serviços que incluem ações, recursos ou condições definidas sem permissões:

a. Avisos de recursos: para serviços que não fornecem permissões para todas as ações ou recursos incluídos, você verá um dos seguintes avisos na coluna Resource (Recurso) da tabela:

   - ⚠ Nenhum recurso é definido. : isso significa que o serviço tem ações definidas, mas nenhum recurso compatível está incluído na política.
   - ⚠ Uma ou mais ações não têm um recurso aplicável. : isso significa que o serviço tem ações definidas, mas que algumas dessas ações não têm um recurso compatível.

-

  Um ou mais recursos não têm uma ação aplicável. : isso significa que o serviço tem recursos definidos, mas que alguns desses recursos não têm uma ação compatível.

Se um serviço incluir tanto ações que não têm nenhum recurso aplicável quanto recursos que não têm nenhum recurso aplicável, apenas o aviso Um ou mais recursos não têm uma ação aplicável será exibido. Isso ocorre porque, quando você visualiza o resumo do serviço, os recursos que não se aplicam a nenhuma ação não são mostrados. Para a ação `ListAllMyBuckets`, essa política inclui o último aviso, pois a ação não é compatível com as permissões no nível do serviço e não é compatível com a chave de condição `s3:x-amz-acl`. Se você corrigir o problema de recurso ou condição, o problema restante aparecerá em um aviso detalhado.

b. Avisos de condição de solicitação: para serviços que não fornecem permissões para todas as condições incluídas, você verá um dos seguintes avisos na coluna Request condition (Condição de solicitação) da tabela:

   -

     Uma ou mais ações não têm uma ação aplicável. : Isso significa que o serviço tem ações definidas, mas que algumas dessas ações não têm uma condição compatível.

   -

     Uma ou mais condições não têm uma ação aplicável. : isso significa que o serviço tem condições definidas, mas que algumas dessas condições não têm uma ação compatível.

c. Múltiplo |

   Uma ou mais ações não têm um recurso aplicável. : a instrução `Deny` para o Amazon S3 inclui mais de um recurso. Ela também inclui mais de uma ação e algumas ações suportam os recursos, algumas não. Para exibir esta política, consulte: the section called “Documento da política JSON SummaryAllElements”. Nesse caso, a política inclui todas as ações do Amazon S3, e somente as ações que podem ser executadas em um bucket ou objeto do bucket são negadas.

d.

   No resources are defined (Nenhum recurso definido): o serviço tem ações definidas, mas nenhum recurso compatível está incluído na política e, portanto, o serviço não fornece permissões. Nesse caso, a política inclui ações do CodeCommit, mas nenhum recurso do CodeCommit.

e. DeploymentGroupName | string como | Todas, região | string como | us-west-2

   |

Uma ou mais ações não têm nenhum recurso aplicável. : o serviço tem uma ação definida e, pelo menos, mais uma ação não tem nenhum recurso compatível.

f. Nenhuma |

uma ou mais condições não têm nenhuma ação aplicável. : o serviço tem, pelo menos, uma chave de condição que não tem nenhuma ação compatível.

Documento da política JSON SummaryAllElements

A política SummaryAllElements não deve ser usada para definir permissões na sua conta. Em vez disso, ela é incluída para demonstrar os erros e avisos que você pode encontrar ao visualizar um resumo de política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
```

```
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-
project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
```

```
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Visualizar resumos de políticas

Você pode visualizar os resumos de políticas para qualquer política anexada a um usuário ou perfil do IAM. Para políticas gerenciadas, você pode visualizar os resumos das políticas na página Políticas. Se a política não incluir um resumo, consulte Resumo de política ausente para saber por quê.

Visualização de resumos de políticas na página Políticas

Visualize o resumo de políticas para políticas gerenciadas na página Políticas.

Para visualizar o resumo da política na página Políticas

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política que deseja visualizar.
4. Na página Detalhes da política, visualize a guia Permissões para ver o resumo da política.

Visualização de um resumo da política para uma política anexada a um usuário

Você pode visualizar o resumo da política para qualquer política anexada a um usuário ou perfil do IAM.

Para visualizar o resumo de uma política anexada a um usuário

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Usuários no painel de navegação.
3. Na lista de usuários, escolha o nome do usuário cuja política deseja visualizar.
4. Na página Resumo do usuário, visualize a guia Permissões para ver a lista de políticas que estão anexadas ao usuário diretamente ou a partir de um grupo.
5. Na tabela de políticas do usuário, expanda a linha da política que deseja visualizar.

Visualização de um resumo da política para uma política anexada a um perfil

Você pode visualizar o resumo da política para qualquer política anexada a um perfil.

Para visualizar o resumo de uma política anexada a uma função

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Na lista de funções, escolha o nome da função cuja política deseja visualizar.
4. Na página Resumo da função, visualize a guia Permissões para ver a lista de políticas que estão anexadas à função.
5. Na tabela de políticas da função, expanda a linha da política que deseja visualizar.

Editar políticas para corrigir avisos

Ao visualizar um resumo da política, você pode encontrar um erro de digitação ou perceber que a política não fornece as permissões esperadas. Não é possível editar o resumo da política diretamente. No entanto, você pode editar uma política gerenciada pelo cliente usando o editor visual de políticas, que detecta muitos dos erros e avisos relatados pelo resumo da política. Em seguida, você pode visualizar as alterações no resumo da política para confirmar que corrigiu todos os problemas. Para saber como editar uma política em linha, consulte the section called “Editar políticas do IAM”. Você não pode editar políticas gerenciadas pela AWS.

Você pode editar uma política para o resumo da política usando a opção Visual.

Para editar uma política para o resumo da política usando a opção Visual

1. Abra o resumo da política conforme explicado nos procedimentos anteriores.
2. Escolha Editar.

   Se estiver na página Usuários e escolher editar uma política gerenciada pelo cliente que esteja anexada a esse usuário, você será redirecionado para a página Políticas. Você pode editar políticas gerenciadas pelo cliente apenas na página Políticas.
3. Escolha a opção Visual para visualizar a representação visual editável da política. O IAM pode reestruturar sua política a fim de otimizá-la para o editor visual e facilitar a identificação e correção de problemas. Os avisos e as mensagens de erro nessa página podem orientar você na correção dos problemas da sua política. Para obter mais informações sobre como o IAM reestrutura políticas, consulte Reestruturação da política.
4. Edite a política e escolha Avançar para ver as alterações refletidas no resumo da política. Se você ainda encontrar um problema, escolha Anterior para retornar à tela de edição.
5. Escolha Salvar alterações para salvar suas alterações.

Você pode editar uma política para o resumo de política usando a opção JSON.

Para editar uma política para o resumo da política usando a opção JSON

1. Abra o resumo da política conforme explicado nos procedimentos anteriores.
2. Você pode usar os botões Resumo e JSON para comparar o resumo da política e o documento da política JSON. Você pode usar essas informações para determinar quais linhas no documento de política que você deseja alterar.
3. Escolha Editar e depois escolha a opção JSON para editar o documento da política JSON.

> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar na opção de editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

Se estiver na página Usuários e escolher editar uma política gerenciada pelo cliente que esteja anexada a esse usuário, você será redirecionado para a página Políticas. Você pode editar políticas gerenciadas pelo cliente apenas na página Políticas.

4. Edite sua política. Resolva os avisos de segurança, as mensagens erros ou os avisos gerais gerados durante a validação de política, e depois escolha Próximo. Se você ainda encontrar um problema, escolha Anterior para retornar à tela de edição.
5. Escolha Salvar alterações para salvar suas alterações.

Níveis de acesso em resumos de política

AWSResumo do nível de acesso da

Resumos de políticas incluem um resumo de nível de acesso que descreve as permissões de ação definidas para cada serviço mencionado na política. Para saber mais sobre os resumos de política, consulte Resumos de políticas. Resumos de nível de acesso indicam se as ações em cada nível de acesso (`List`, `Read`, `Tagging`, `Write` e `Permissions management`) têm permissões `Full` ou `Limited` definidas na política. Para visualizar a classificação de nível de acesso atribuída a cada ação em um serviço, consulte Ações, recursos e chaves de condição de serviços da AWS.

O exemplo a seguir descreve o acesso fornecido por uma política para os serviços oferecidos. Para obter exemplos de documentos de políticas JSON completos e seus resumos relacionados, consulte Exemplos de resumos de políticas.

Serviço	Nível de acesso	Essa política fornece o seguinte
IAM	Acesso total.	Acesso a todas as ações dentro do serviço IAM.
CloudWatch	Completo: Listar	Acesso a todas as ações do CloudWatch no nível de acesso `List`, mas nenhum acesso a ações com classificação de nível de acesso `Read`, `Write` ou `Permissions management` .
Data Pipeline	Limitado: Listar, Leir	Acesso a, pelo menos, uma ação do AWS Data Pipeline, mas não todas, no nível de acesso

Serviço	Nível de acesso	Essa política fornece o seguinte
		`List` e `Read`, mas não às ações `Write` ou `Permissions management` .
EC2	Completo: Listar, Ler Limitado: Gravar	Acesso a todas as ações `List` and `Read` do Amazon EC2 e acesso a pelo menos uma, mas não a todas as ações `Write` do Amazon EC2, mas nenhum acesso às ações com a classificação de nível de acesso `Permissions management` .
S3	Limitado: Ler, Gravar, Gerenciamento de permissões	Acesso a pelo menos uma, mas não todas as ações `Read`, `Write` e `Permissions management` do Amazon S3.
codedploy	(empty)	Acesso desconhecido, pois o IAM não reconhece este serviço.
API Gateway	Nenhum	Nenhum acesso é definido na política.
CodeBuild	⚠ Nenhuma ação é definida.	Sem acesso porque nenhuma ação é definida para o serviço. Para saber como entender e resolver esse problema, consulte the section called “Minha política não concede as permissões esperadas”.

Em um resumo de política, Acesso completo indica que a política fornece acesso a todas as ações do serviço. Políticas que fornecem acesso a algumas, mas não a todas as ações de um serviço também são agrupadas de acordo com a classificação de nível de acesso. Isso é indicado por um dos seguintes agrupamentos de nível de acesso:

- Completo: a política fornece acesso a todas as ações na classificação de nível de acesso especificada.
- Limitado: a política fornece acesso a uma ou mais, mas não a todas as ações na classificação de nível de acesso especificada.

- Nenhum: a política não fornece acesso.
- (vazio): o IAM não reconhece este serviço. Se o nome do serviço inclui um erro ortográfico, a política não fornece acesso ao serviço. Se o nome do serviço está correto, talvez o serviço possa não dar suporte aos resumos de políticas ou possa estar em pré-visualização. Nesse caso, a política pode oferecer acesso, mas este acesso não pode ser mostrado no resumo de política. Para solicitar o suporte do resumo da política para uma serviço disponível, consulte O serviço não oferece suporte a resumos de política do IAM.

Resumos de nível de acesso que incluem acesso limitado (parcial) às ações são agrupados usando as classificações de nível de acesso AWS, `List`, `Read`, `Tagging` ou `Write` da `Permissions management`.

AWSNíveis de acesso da

AWSA define as seguintes classificações de nível de acesso para as ações em um serviço:

- Listar: Permissão para listar recursos dentro do serviço para determinar se um objeto existe. Ações com esse nível de acesso podem listar objetos, mas não podem ver os conteúdos de um recurso. Por exemplo, a ação `ListBucket` do Amazon S3 tem o nível de acesso List (Lista).
- Ler: Permissão para ler, mas não editar os conteúdos e atributos de recursos no serviço. Por exemplo, as ações `GetObject` e `GetBucketLocation` do Amazon S3 têm o nível de acesso Read (Leitura).
- Marcação: permissão para executar ações que apenas alteram o estado de tags de recurso. Por exemplo, as ações do IAM `TagRole` e `UntagRole` têm o nível de acesso Tagging (Etiquetamento) porque permitem apenas etiquetar ou desetiquetar uma função. No entanto, a ação `CreateRole` permite marcar um recurso de função quando você criar essa função. Como a ação não apenas adiciona uma tag, ela tem o nível de acesso `Write`.
- Gravar: permissão para criar, excluir ou modificar recursos no serviço. Por exemplo, as ações `CreateBucket`, `DeleteBucket` e `PutObject` do Amazon S3 têm o nível de acesso Write (Gravação). As ações de `Write` (gravação) também podem permitir a modificação de uma etiqueta de recurso. No entanto, uma ação que permite apenas alterações nas tags tem o nível de acesso `Tagging`.
- Gerenciamento de permissões: gerenciamento de permissões se refere às ações que controlam o acesso nos Serviços da AWS, incluindo as permissões de identidades, do IAM ou não, mas exclui os controles de acesso no nível da rede, como os grupos de segurança. Por exemplo, a

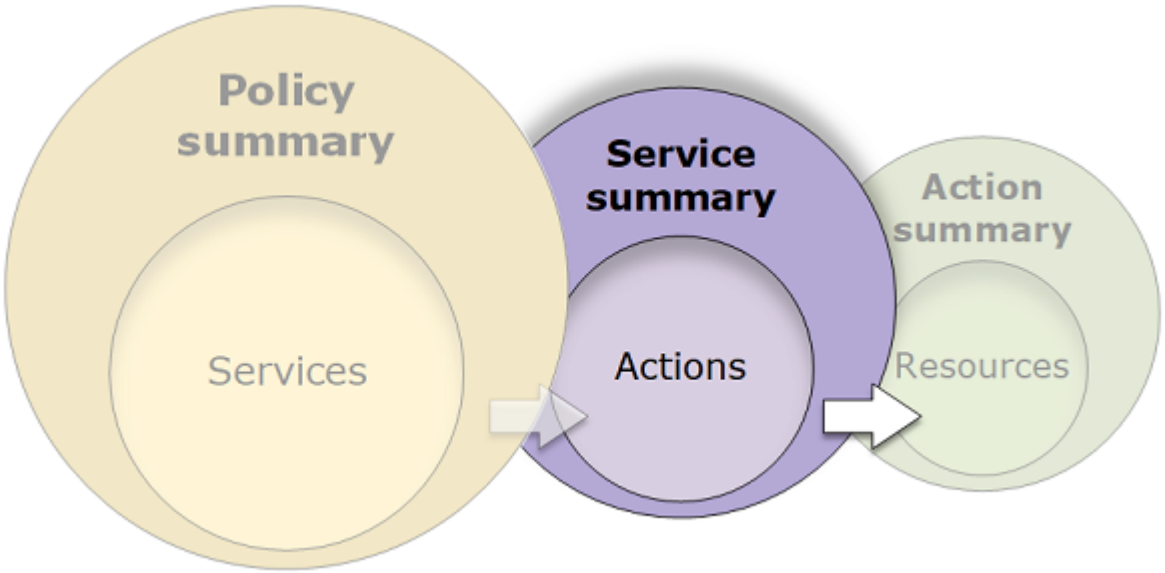
maioria das ações do IAM e do AWS Organizations, bem como as ações `PutBucketPolicy` e `DeleteBucketPolicy` do Amazon S3, têm o nível de acesso de Gerenciamento de permissões.

> **Dica**
>
> Para melhorar a segurança da Conta da AWS, restrinja ou monitore regularmente políticas que incluam a classificação de nível de acesso Gerenciamento de permissões.

Para visualizar a classificação do nível de acesso para todas as ações em um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS.

Resumo do serviço (lista de ações)

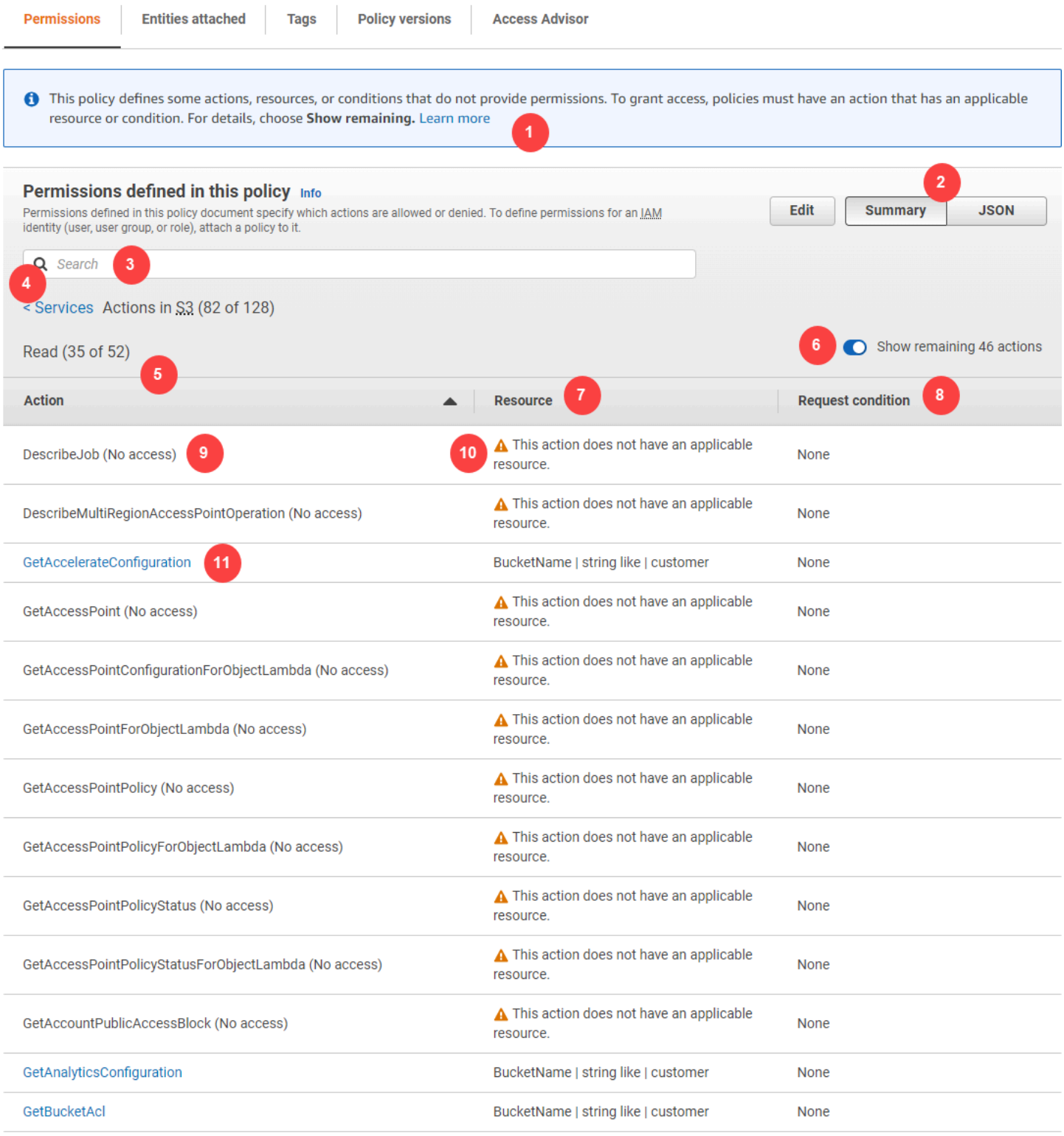
As políticas são resumidas em três tabelas: o resumo de políticas, o resumo de serviços e o resumo de ações. A tabela do resumo de serviços inclui uma lista das ações e resumos das permissões que são definidas pela política para o dado serviço.



Visualize um resumo de serviços para cada serviço listado no resumo de políticas que concede permissões. A tabela é agrupada em Ações não categorizadas, Tipos de recursos não categorizados e seções de nível de acesso. Se a política incluir uma ação que o IAM não reconhece, a ação será incluída na seção Uncategorized actions (Ações não categorizadas) da tabela. Se o IAM reconhecer a ação, ela será incluída em uma das seções de nível de acesso (Listar, Ler, Gravar e Gerenciamento de permissões) da tabela. Para visualizar a classificação de nível de acesso atribuída a cada ação em um serviço, consulte Ações, recursos e chaves de condição de serviços da AWS.

Noções básicas sobre os elementos de um resumo do serviço

O exemplo a seguir é o resumo do serviço para as ações do Amazon S3 que são permitidas com base no resumo da política. As ações para esse serviço estão agrupados por nível de acesso. Por exemplo, 35 ações de Leitura estão definidas do total de 52 ações de Leitura disponíveis para o serviço.



A página do resumo de serviços de uma política gerenciada inclui as seguintes informações:

1. Se a política não conceder permissões para todas as ações, recursos e condições definidos para o serviço na política, um banner de aviso aparecerá na parte superior da página. O resumo de

serviços inclui detalhes sobre o problema. Para saber como os resumos de política ajudam a entender e solucionar problemas das permissões concedidas por sua política, consulte the section called “Minha política não concede as permissões esperadas”.

2. Escolha JSON para ver detalhes adicionais sobre a política. Faça isso para visualizar todas as condições que são aplicadas às ações. (Se você estiver visualizando o resumo de serviços para uma política em linha que esteja anexado diretamente a um usuário, será necessário fechar a caixa de diálogo do resumo de serviços e voltar para o resumo de políticas para acessar o documento de política JSON.)
3. Para visualizar o resumo para uma ação específica, digite palavras-chave na caixa Pesquisar para reduzir a lista de ações disponíveis.
4. Ao lado da seta invertida Serviços aparece o nome do serviço (neste caso, S3). O resumo do serviço para esse serviço inclui a lista de ações permitidas ou negadas que são definidas na política. Se o serviço aparecer sob (Negação explícita) na guia Permissões, as ações listadas na tabela de resumo do serviço serão explicitamente negadas. Se o serviço aparecer sob Permitir na guia Permissões, as ações listadas na tabela de resumo do serviço serão explicitamente permitidas.
5. Ação: essa coluna lista as ações que são definidas na política e fornece os recursos e as condições para cada ação. Se a política conceder ou negar permissões para a ação, o nome da ação será vinculado à tabela resumo das ações. A tabela agrupa essas ações em pelo menos uma e até quatro seções, dependendo do nível de acesso que a política permite ou nega. As seções são Lista, Leitura, Gravação, Gerenciamento de permissões e Marcação. A contagem indica o número de ações reconhecidas que fornecem permissões em cada nível de acesso. O total é o número de ações conhecidas do serviço. Neste exemplo, 35 ações, de um total de 52 ações, fornecem permissões de Leitura conhecidas do Amazon S3. Para visualizar a classificação de nível de acesso atribuída a cada ação em um serviço, consulte Ações, recursos e chaves de condição de serviços da AWS.
6. Mostrar ações restantes: alterne este botão para expandir ou ocultar a tabela para incluir ações que são conhecidas, mas que não fornecem permissões para esse serviço. Alternar o botão também exibe avisos para os elementos que não fornecem permissões.
7. Resource (Recurso): esta coluna mostra os recursos que a política define para o serviço. O IAM não verifica se o recurso se aplica a cada ação. Neste exemplo, as ações do serviço Amazon S3 só são permitidas no recurso `developer_bucket` de bucket do Amazon S3. Dependendo das informações que o serviço fornece ao IAM, talvez seja exibido um ARN, como `arn:aws:s3:::developer_bucket/*`, ou um tipo de recurso definido, como `BucketName = developer_bucket`.

> **Note**
>
> Essa coluna pode incluir um recurso de um serviço diferente. Se a declaração de política que inclui o recurso não incluir as duas ações e recursos do mesmo serviço, a política incluirá recursos incompatíveis. O IAM não avisa sobre recursos incompatíveis quando você cria uma política, ou quando você visualiza uma política no resumo do serviço. O IAM também não indica se a ação se aplica aos recursos, mas apenas se o serviço é correspondente. Se essa coluna incluir um recurso incompatível, será necessário analisar sua política para busca de erros. Para compreender melhor suas políticas, sempre teste-as com o simulador de políticas.

8. Request condition (Condição de solicitação): esta coluna mostra se as ações associadas ao recurso estão sujeitas a condições. Para saber mais sobre essas condições, escolha JSON para revisar o documento de política JSON.
9. (No access) (Sem acesso): esta política inclui uma ação que não fornece permissões.
10. Resource warning (Aviso do recurso): para ações com recursos que não fornecem permissões completas, você verá um dos seguintes avisos:
    - Esta ação não é compatível com permissões no nível do recurso. Isso requer um caractere curinga (*) para o recurso. : isso significa que a política inclui as permissões no nível de recurso, mas deve incluir `"Resource": ["*"]` para fornecer permissões para esta ação.
    - Esta ação não tem um recurso aplicável. : isso significa que a ação é incluída na política sem um recurso compatível.
    - Esta ação não tem um recurso e uma condição aplicáveis. : isso significa que a ação é incluída na política sem um recurso e sem uma condição compatíveis. Neste caso, há também uma condição incluída na política para esse serviço, mas não há condições que se aplicam a esta ação.
11. Ações que fornecem permissões incluem um link para o resumo da ação.

Visualizar resumos de serviços

Visualize um resumo de serviços para cada serviço listado no resumo de políticas que concede permissões.

Visualização de resumos de serviço na página Políticas

Você pode visualizar o resumo do serviço para políticas gerenciadas na página Políticas.

Para visualizar o resumo de serviços de uma política gerenciada

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política que deseja visualizar.
4. Na página Detalhes da política, visualize a guia Permissões para ver o resumo da política.
5. Na lista de serviços do resumo de políticas, escolha o nome do serviço que deseja visualizar.

Para visualizar um resumo do serviço para uma política anexada a um usuário

Você pode visualizar o resumo do serviço para qualquer política anexada a um usuário do IAM.

Para visualizar o resumo de serviços de uma política anexada a um usuário

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Na lista de usuários, escolha o nome do usuário cuja política deseja visualizar.
4. Na página Resumo do usuário, visualize a guia Permissões para ver a lista de políticas que estão anexadas ao usuário diretamente ou a partir de um grupo.
5. Na tabela de políticas do usuário, escolha o nome da política que deseja visualizar.

   Se estiver na página Usuários e escolher visualizar o resumo do serviço para uma política gerenciada pelo cliente anexada a esse usuário, você será redirecionado para a página Políticas. Você só pode visualizar os resumos dos serviços na página Políticas.
6. Escolha Resumo. Na lista de serviços do resumo de políticas, escolha o nome do serviço que deseja visualizar.

> **Note**
>
> Se a política que você selecionar for uma política em linha que está anexada diretamente ao usuário, a tabela do resumo de serviços será exibida. Se a política for uma política em linha anexada a um grupo, você será levado para o documento de política JSON desse grupo. Se a política for uma política gerenciada, você será levado para o resumo de serviços dessa política na página Políticas.

Visualização de um resumo do serviço para uma política anexada a um perfil

Você pode visualizar o resumo da política para qualquer política anexada a um perfil.

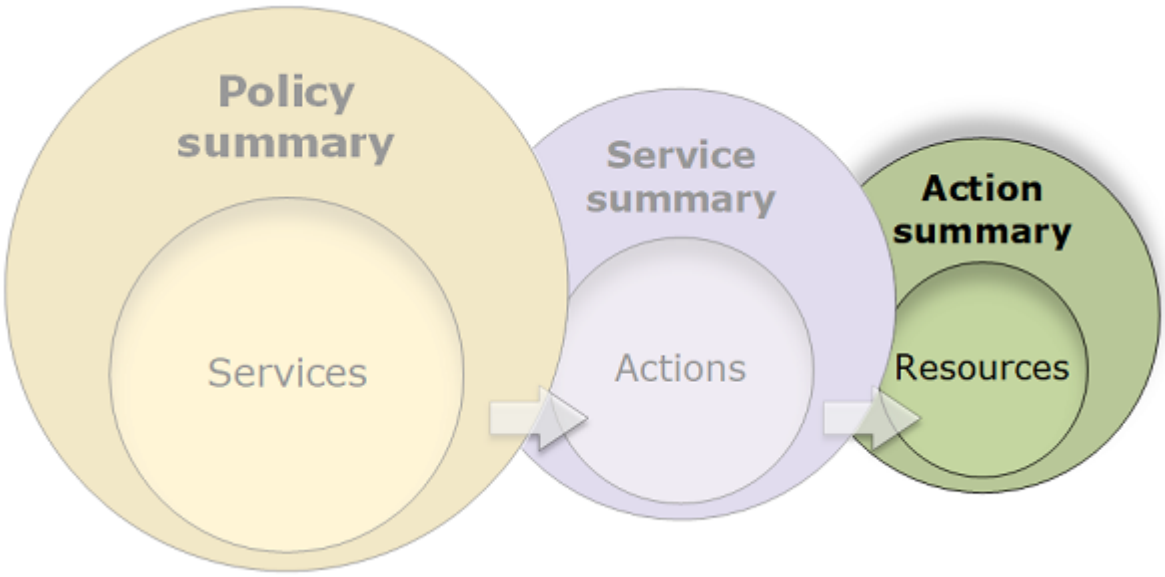
Para visualizar o resumo de serviços de uma política anexada a uma função

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Funções no painel de navegação.
3. Na lista de funções, escolha o nome da função cuja política deseja visualizar.
4. Na página Resumo da função, visualize a guia Permissões para ver a lista de políticas que estão anexadas à função.
5. Na tabela de políticas para o perfil, escolha o nome da política que você deseja visualizar.

   Se estiver na página Perfis e escolher visualizar um resumo do serviço para uma política anexada a esse usuário, você será redirecionado para a página Políticas. Você só pode visualizar os resumos dos serviços na página Políticas.
6. Na lista de serviços do resumo de políticas, escolha o nome do serviço que deseja visualizar.

Resumo da ação (lista de recursos)

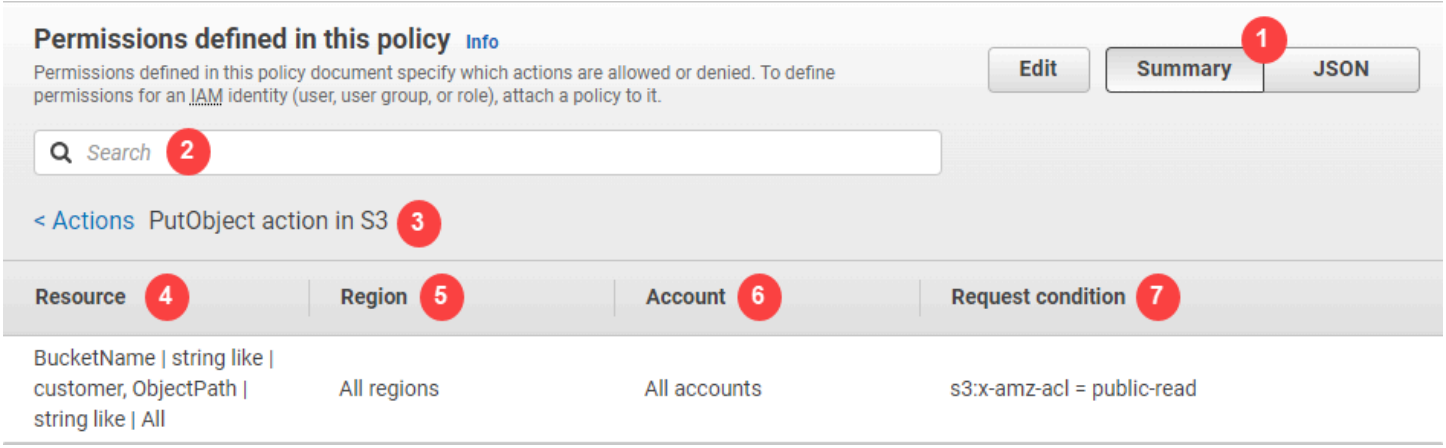
As políticas são resumidas em três tabelas: o resumo de políticas, o resumo de serviços e o resumo de ações. A tabela resumo da ação inclui uma lista de recursos e as condições associadas que se aplicam à ação escolhida.



Para ver um resumo para cada ação que concede permissões, escolha o link no resumo do serviço. A tabela de resumo da ação inclui detalhes sobre o recurso, incluindo a região e a conta. Você também pode visualizar as condições que se aplicam a cada recurso. Isso mostra as condições que se aplicam a alguns recursos, mas não a outros.

Noções básicas sobre os elementos de um resumo da ação

O exemplo a seguir é do resumo da ação `PutObject` (Gravar) do resumo do serviço Amazon S3 (consulte Resumo do serviço (lista de ações)). Para essa ação, a política define várias condições em um único recurso.



A página de resumo de ações inclui as seguintes informações:

1. Escolha JSON para ver detalhes adicionais sobre a política, como as várias condições que se aplicam às ações. (Se você estiver visualizando o resumo da ação para uma política em linha anexada diretamente a um usuário, as etapas serão diferentes. Para acessar o documento da política JSON nesse caso, você deve fechar a caixa de diálogo do resumo da ação e retornar ao resumo da política.)
2. Para visualizar o resumo para um recurso específico, digite palavras-chave na caixa Pesquisar para reduzir a lista de recursos disponíveis.
3. Ao lado da seta invertida Ações é exibido o nome do serviço e a ação no formato `action name action in service` (nesse caso, Ação PutObject no S3). O resumo da ação desse serviço inclui a lista de recursos que estão definidos na política.
4. Resource (Recurso): esta coluna lista os recursos que a política define para o serviço escolhido. Neste exemplo, a ação PutObject é permitida em todos os caminhos de objeto, mas somente no recurso de bucket `developer_bucket` Amazon S3. Dependendo das informações que o serviço fornece ao IAM, talvez seja exibido um ARN, como `arn:aws:s3:::developer_bucket/*`,

ou um tipo de recurso definido, como `BucketName = developer_bucket, ObjectPath = All`.

5. Region (Região): esta coluna mostra a região em que o recurso é definido. Os recursos podem ser definidos para todas as regiões ou a uma única região. Eles não podem existir em mais de uma região específica.
   - Todas as regiões: as ações associadas ao recurso se aplicam a todas as regiões. Neste exemplo, a ação pertence a um serviço global, Amazon S3. Ações que pertencem a serviços globais se aplicam a todas as regiões.
   - Texto da região: as ações associadas ao recurso se aplicam a uma região. Por exemplo, uma política pode especificar a região `us-east-2` para um recurso.
6. Account (Conta): esta coluna indica se os serviços ou as ações associados ao recurso se aplicam a uma conta específica. Os recursos podem existir em todas as contas ou em uma única conta. Eles não podem existir em mais de uma conta específica.
   - All accounts (Todas as contas): as ações que estão associadas ao recurso se aplicam a todas as contas. Neste exemplo, a ação pertence a um serviço global, Amazon S3. Ações que pertencem a serviços globais se aplicam a todas as contas.
   - Esta conta: as ações associadas ao recurso só se aplicam à conta atual.
   - Número da conta: as ações associadas ao recurso aplicam-se somente a uma conta (na qual você não está registrado no momento). Por exemplo, se uma política especificar a conta `123456789012` para um recurso, o número da conta será exibido no resumo da política.
7. Request condition (Condição de solicitação): esta coluna mostra se as ações associadas ao recurso estão sujeitas a condições. Este exemplo inclui a condição `s3:x-amz-acl = public-read`. Para saber mais sobre essas condições, escolha JSON para revisar o documento de política JSON.

Visualizar resumos de ações

Você pode visualizar um resumo da ação para cada ação listada no resumo da política que concede permissões.

Visualização de resumos de ações na página Políticas

Você pode visualizar o resumo da ação para políticas gerenciadas.

Para visualizar o resumo da ação de uma política gerenciada

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.
3. Na lista de políticas, escolha o nome da política que deseja visualizar.
4. Na página Detalhes da política, visualize a guia Permissões para ver o resumo da política.
5. Na lista de serviços do resumo de políticas, escolha o nome do serviço que deseja visualizar.
6. Na lista de resumo do serviço da ação, escolha o nome da ação que deseja visualizar.

Visualização de resumos de ações para uma política anexada a um usuário

Você pode visualizar o resumo da ação para qualquer política anexada a um usuário.

Para visualizar o resumo da ação de uma política anexada a um usuário

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Usuários no painel de navegação.
3. Na lista de usuários, escolha o nome do usuário cuja política deseja visualizar.
4. Na página Resumo do usuário, visualize a guia Permissões para ver a lista de políticas que estão anexadas ao usuário diretamente ou a partir de um grupo.
5. Na tabela de políticas do usuário, escolha o nome da política que deseja visualizar.

   Se estiver na página Usuários e escolher visualizar o resumo do serviço para uma política gerenciada pelo cliente anexada a esse usuário, você será redirecionado para a página Políticas. Você só pode visualizar os resumos dos serviços na página Políticas.
6. Na lista de serviços do resumo de políticas, escolha o nome do serviço que deseja visualizar.

> **Note**
>
> Se a política que você selecionar for uma política em linha que está anexada diretamente ao usuário, a tabela do resumo de serviços será exibida. Se a política for uma política em linha anexada a um grupo, você será levado para o documento de política JSON desse grupo. Se a política for uma política gerenciada, você será levado para o resumo de serviços dessa política na página Políticas.

7. Na lista de resumo do serviço da ação, escolha o nome da ação que deseja visualizar.

Visualização de resumos de ações para uma política anexada a um perfil

Você pode visualizar o resumo da ação para qualquer política anexada a um perfil.

Para visualizar o resumo da ação de uma política anexada a uma função

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Na lista de funções, escolha o nome da função cuja política deseja visualizar.
4. Na página Resumo da função, visualize a guia Permissões para ver a lista de políticas que estão anexadas à função.
5. Na tabela de políticas para o perfil, escolha o nome da política que você deseja visualizar.

   Se estiver na página Perfis e escolher visualizar um resumo do serviço para uma política anexada a esse usuário, você será redirecionado para a página Políticas. Você só pode visualizar os resumos dos serviços na página Políticas.
6. Na lista de serviços do resumo de políticas, escolha o nome do serviço que deseja visualizar.
7. Na lista de resumo do serviço da ação, escolha o nome da ação que deseja visualizar.

Exemplos de resumos de políticas

Os exemplos a seguir incluem políticas JSON com seus resumos de políticas associados, os resumos de serviços e os resumos de ações para ajudá-lo a compreender as permissões dadas por meio de uma política.

Política 1: DenyCustomerBucket

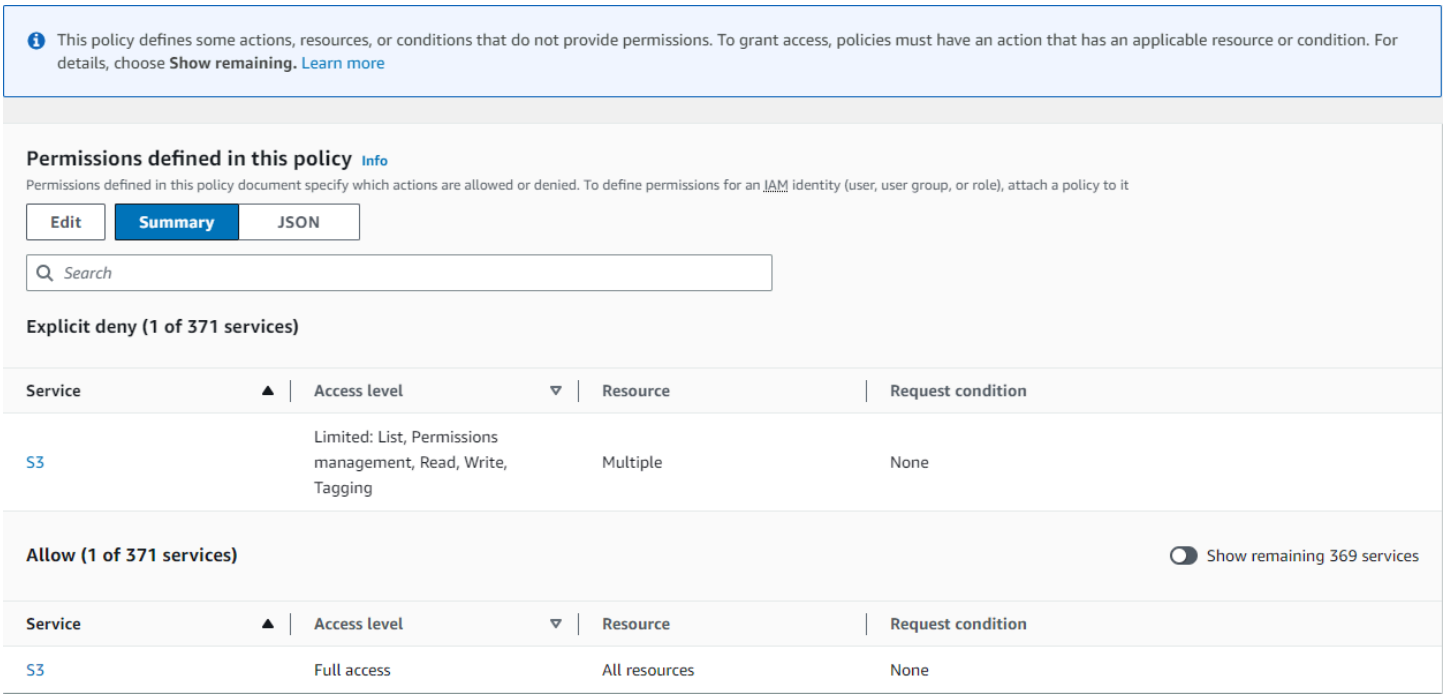
Essa política demonstra uma permissão e negação para o mesmo serviço.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucket Resumo de políticas:



Resumo do serviço DenyCustomerBucket S3 (Negação explícita):

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject (Ler) Resumo da ação:

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

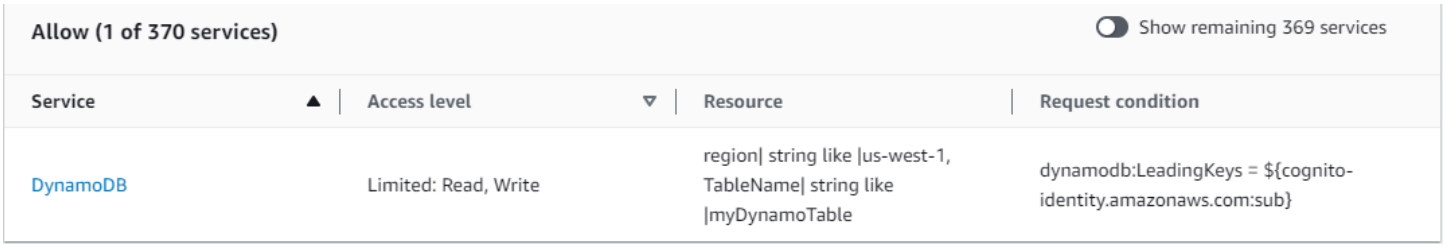
Política 2: DynamoDbRowCognitoID

Esta política fornece acesso no nível de linha ao Amazon DynamoDB com base no ID do Amazon Cognito do usuário.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

Resumo da política DynamoDbRowCognitoID:



Resumo do serviço DynamoDbRowCognitoID DynamoDB (Permitir):

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Resumo da ação GetItem (Listar):

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Política 3: MultipleResourceCondition

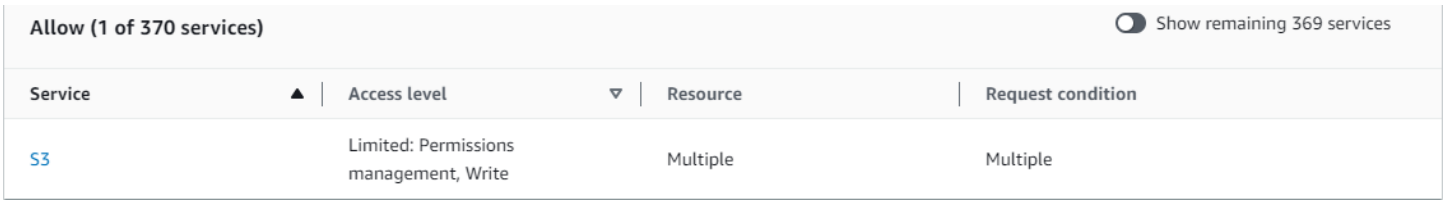
Essa política inclui vários recursos e condições.

JSON

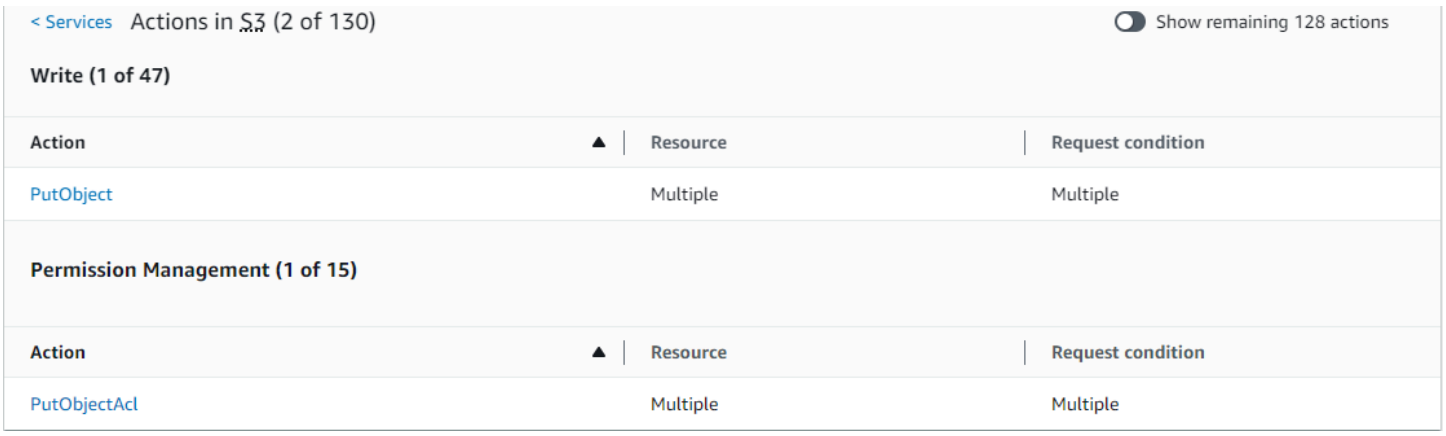
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

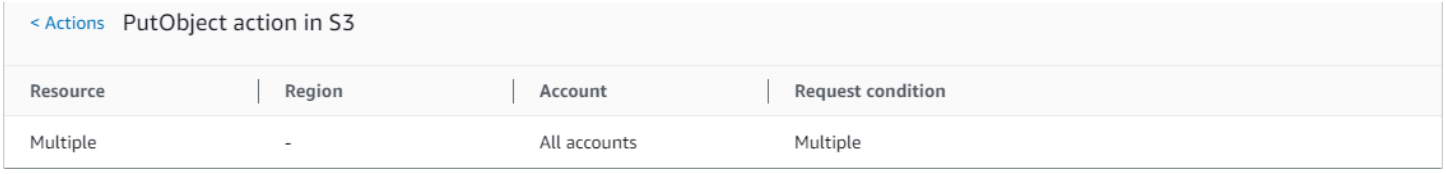
Resumo da política MultipleResourceCondition:



Resumo do serviço MultipleResourceCondition S3 (Permitir):



Resumo da ação PutObject (Gravar):



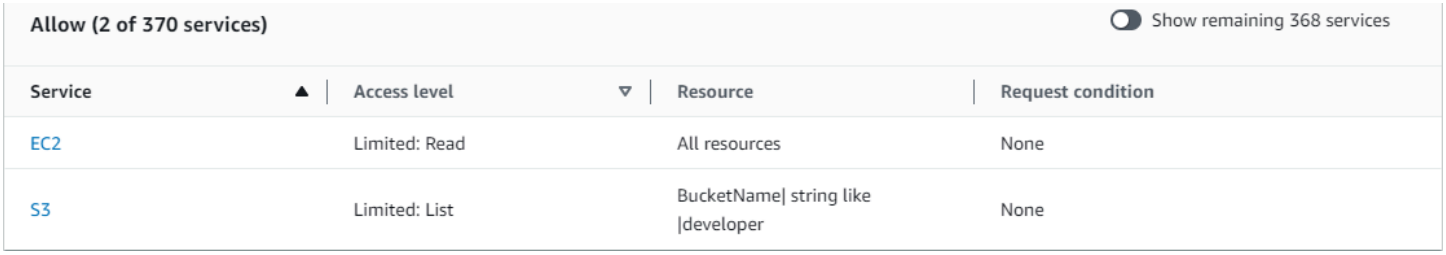
Política 4: EC2_Troubleshoot

A política a seguir permite que os usuários obtenham uma captura de tela de uma instância do Amazon EC2 em execução, o que pode ajudar a solucionar problemas com o EC2. Essa política também permite visualizar informações sobre os itens no bucket do desenvolvedor do Amazon S3.

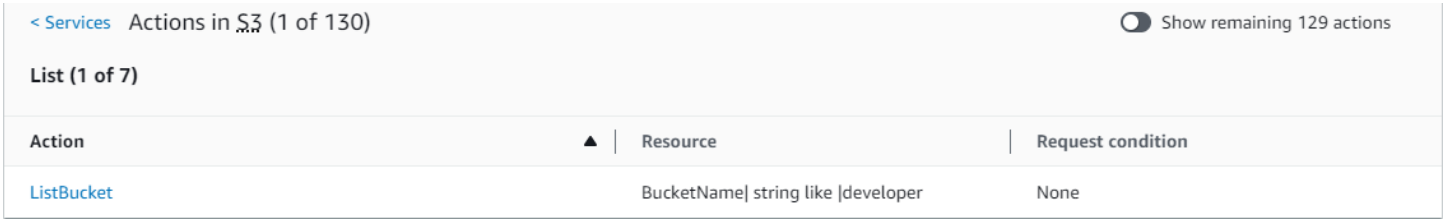
JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

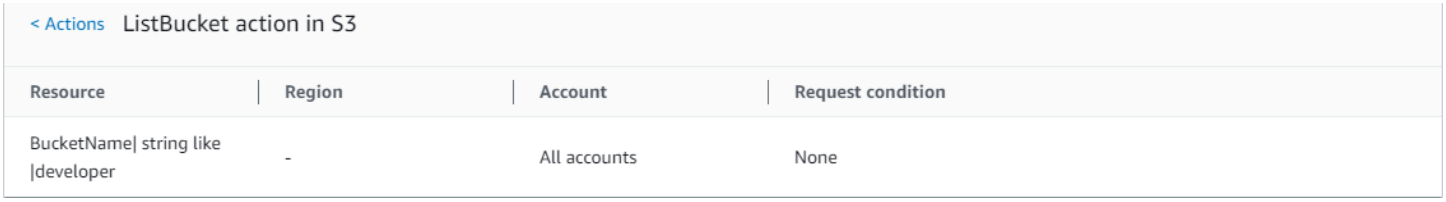
Resumo da política EC2_Troubleshoot:



Resumo do serviço EC2_Troubleshoot S3 (Permitir):



Resumo da ação ListBucket (Listar):



Política 5: CodeBuild_CodeCommit_CodeDeploy

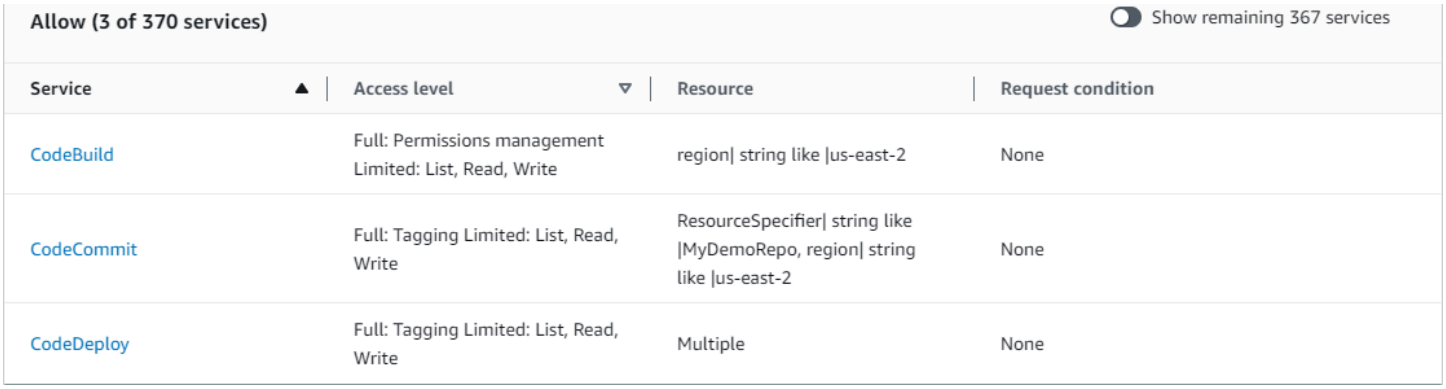
Essa política fornece acesso aos recursos CodeCommit, CodeDeploy e CodeBuild específicos. Como esses recursos são específicos para cada serviço, eles aparecem apenas com o serviço correspondente. Se você incluir um recurso que não corresponde a nenhum dos serviços no elemento `Action`, o recurso será exibido em todos os resumos de ações.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-
project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-
east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
```

```
            ]
        }
    ]
}
```

Resumo da política CodeBuild_CodeCommit_CodeDeploy:



Allow (3 of 370 services) — Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

Resumo do serviço CodeBuild_CodeCommit_CodeDeploy CodeBuild (Permitir):

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

Resumo da ação CodeBuild_CodeCommit_CodeDeploy StartBuild (Gravar):

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Permissões necessárias para acessar recursos do IAM

Recursos são objetos dentro de um serviço. Os recursos do IAM incluem grupos, usuários, funções e políticas. Se você fizer login com credenciais de Usuário raiz da conta da AWS, não terá restrições para administrar credenciais do IAM ou recursos do IAM. No entanto, os usuários do IAM devem receber explicitamente permissões para administrar credenciais ou recursos do IAM. Você pode fazer isso anexando uma política baseada em identidade ao usuário.

> **Note**
>
> Em toda a documentação da AWS, quando nos referirmos a uma política do IAM sem mencionar uma das categorias específicas, estaremos nos referindo a uma política baseada em identidade e gerenciada pelo cliente. Para obter detalhes sobre as categorias de políticas, consulte the section called “Políticas e permissões”.

Permissões para administração de identidades do IAM

As permissões necessárias para administrar grupos, usuários, funções e credenciais do IAM geralmente correspondem às ações da API para a tarefa. Por exemplo, para criar usuários do IAM, você deve ter a permissão `iam:CreateUser` que possui o comando de API correspondente: `CreateUser`. Para permitir que um usuário do IAM crie outros usuários do IAM, você pode anexar uma política do IAM, conforme mostrado a seguir, ao usuário em questão:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Em uma política, o valor do elemento `Resource` depende da ação e quais recursos ela pode afetar. No exemplo anterior, a política permite que um usuário crie qualquer usuário (* é um

curinga que corresponde a todas as strings). Por outro lado, uma política que permite que os usuários alterem apenas as próprias chaves de acesso (ações das APIs `CreateAccessKey` e `UpdateAccessKey`) geralmente tem um elemento `Resource`. Neste caso, o ARN inclui uma variável (`${aws:username}`) que é substituída pelo nome do usuário atual, como no exemplo a seguir:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

No exemplo anterior, `${aws:username}` é uma variável que define o nome do usuário atual. Para obter mais informações sobre variáveis de política, consulte Elementos de política do IAM: variáveis e etiquetas.

Usando um caractere curinga (*) no nome da ação, muitas vezes, facilita a concessão de permissões para todas as ações relacionadas a uma tarefa específica. Por exemplo, para permitir que os usuários realizem qualquer ação do IAM, você pode usar `iam:*` para a ação. Para permitir que os usuários realizem qualquer ação relacionada apenas a chaves de acesso, você pode usar `iam:*AccessKey*` no elemento `Action` de uma declaração de política. Isso

dá ao usuário permissão para executar as ações `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`, `ListAccessKeys` e `UpdateAccessKey`. (Se, no futuro, for adicionada uma ação ao IAM que tenha "AccessKey" no nome, o uso de `iam:*AccessKey*` para o elemento `Action` também fornecerá ao usuário permissão para essa nova ação.) O exemplo a seguir mostra uma política que permite que os usuários executem todas as ações relativas às suas próprias chaves de acesso (substitua *account-id* pelo ID da sua Conta da AWS):

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/${aws:username}"
    }
}
```

Algumas tarefas, como a exclusão de um grupo, envolvem várias ações: você deve primeiro remover os usuários do grupo e, em seguida, desanexar ou excluir as políticas do grupo de depois realmente excluir o grupo. Se você deseja que um usuário exclua um grupo, deve oferecer a ele permissões para executar todas as ações relacionadas.

Permissões para trabalhar no Console de gerenciamento da AWS

Os exemplos anteriores mostram políticas que permitem a um usuário executar as ações com a AWS CLI ou os SDKs da AWS.

Conforme os usuários trabalham com o console, o console emite solicitações ao IAM para listar grupos, usuários, funções e políticas e obter as políticas associadas a um grupo, usuário ou função. O console também emite solicitações para obter informações da Conta da AWS e informações sobre a entidade principal. A entidade principal é o usuário que faz as solicitações no console.

Em geral, para realizar uma ação, você deve ter somente a ação correspondente incluída em uma política. Para criar um usuário, você precisa de permissão para chamar a ação `CreateUser`. Muitas vezes, quando você usa o console para executar uma ação, precisa ter permissões para exibir, listar, obter ou visualizar os recursos no console. Isso é necessário para que você possa navegar através do console para realizar a ação especificada. Por exemplo, se o usuário Jorge quiser usar o console

para alterar suas próprias chaves de acesso, ele acessará o console do IAM e escolherá Users (Usuários). Essa ação faz com que o console faça uma solicitação `ListUsers`. Se o Jorge não tiver permissão para a ação `iam:ListUsers`, o console terá o acesso negado ao tentar listar os usuários. Como resultado, Jorge não poderá acessar seu próprio nome e chaves de acesso, mesmo que ele tenha permissões para as ações `CreateAccessKey` e `UpdateAccessKey`.

Se você deseja fornecer aos usuários permissões para administrar usuários, grupos, funções, políticas e credenciais com o Console de gerenciamento da AWS, é necessário incluir permissões para as ações executadas pelo console. Para obter alguns exemplos de políticas que você pode usar para conceder essas permissões a um usuário, consulte Exemplos de política para administrar recursos do IAM.

Conceder permissões entre contas da AWS

Você pode conceder diretamente aos usuários do IAM em sua própria conta acesso aos seus recursos. Se os usuários de outra conta precisarem de acesso aos seus recursos, você poderá criar uma função do IAM, que é uma entidade que inclui permissões, mas que não está associada a um usuário específico. Os usuários de outras contas podem, então, usar a função e acessar os recursos de acordo com as permissões que você tiver atribuído à função. Para obter mais informações, consulte Acesso a um usuário do IAM em outra Conta da AWS de sua propriedade.

> **Note**
>
> Alguns serviços oferecem suporte a políticas baseadas em recurso, conforme descrito em Políticas baseadas em identidade e em recurso (como o Amazon S3, Amazon SNS e Amazon SQS). Para esses serviços, uma alternativa ao uso de funções é anexar uma política ao recurso (bucket, tópico ou fila) que você deseja compartilhar. A política baseada em recurso pode especificar a conta da AWS com permissões para acessar o recurso.

Permissões para um serviço acessar outro

Muitos serviços da AWS acessam outros serviços da AWS. Por exemplo, vários produtos da AWS: incluindo o Amazon EMR, o Elastic Load Balancing e o Amazon EC2 Auto Scaling, gerenciam instâncias do Amazon EC2. Outros produtos da AWS usam buckets do Amazon S3, tópicos do Amazon SNS, filas do Amazon SQS e assim por diante.

O cenário para o gerenciamento de permissões nesses casos varia de acordo com o serviço. Veja a seguir alguns exemplos de como as permissões são tratadas para diferentes serviços:

- No Amazon EC2 Auto Scaling, os usuários devem ter permissão para usar o Auto Scaling, mas não precisam receber permissão explicitamente para gerenciar instâncias do Amazon EC2.
- No AWS Data Pipeline, uma função do IAM determina o que um pipeline pode fazer; os usuários precisam de permissão para assumir a função. (Para obter mais detalhes, consulte Conceder permissões a pipelines com o IAM no Guia do desenvolvedor do AWS Data Pipeline.)

Para obter detalhes sobre como configurar permissões corretamente para que um serviço da AWS seja capaz de realizar as tarefas pretendidas, consulte a documentação do serviço que você está chamando. Para saber como criar uma função para um serviço, consulte Criar um perfil para delegar permissões a um serviço da AWS.

Configuração de um serviço com uma função do IAM para trabalhar em seu nome

Quando quiser configurar um produto da AWS para trabalhar em seu nome, normalmente você fornecerá o ARN para uma função do IAM que define o que o serviço tem permissão para fazer. A AWS verifica para garantir que você tenha permissões para passar uma função para um serviço. Para obter mais informações, consulte Conceda permissões a um usuário para passar um perfil para um serviço da AWS.

Ações necessárias

As ações são as atividades que você pode realizar com um recurso, como visualizar, criar, editar e excluir esse recurso. As ações são definidas por cada serviço da AWS.

Para permitir que alguém realize uma ação, você deve incluir as ações necessárias em uma política que se aplique à identidade da chamada ou ao recurso afetado. Em geral, para fornecer a permissão necessária para realizar uma ação, você deve incluir essa ação em sua política. Por exemplo, para criar um usuário, você precisa adicionar a ação CreateUser à política.

Em alguns casos, uma ação pode exigir que você inclua ações adicionais relacionadas em sua política. Por exemplo, para fornecer permissão para que uma pessoa crie um diretório no AWS Directory Service usando a operação `ds:CreateDirectory`, você deve incluir as seguintes ações na política dela:

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`

- `ec2:CreateSecurityGroup`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Ao criar ou editar uma política usando o editor visual, você recebe avisos e solicitações para ajudá-lo a escolher todas as ações necessárias para a política.

Para obter mais informações sobre as permissões necessárias para criar um diretório no AWS Directory Service, consulte Exemplo 2: permitir que um usuário crie um diretório.

Exemplos de política para administrar recursos do IAM

Veja seguir exemplos de políticas do IAM que permitem aos usuários executar tarefas associadas ao gerenciamento de usuários, grupos e credenciais do IAM. Isso inclui políticas que permitem que os usuários gerenciem as próprias senhas, chaves de acesso e dispositivos de autenticação multifator (MFA).

Para obter exemplos de políticas que permitem aos usuários realizar tarefas com outros produtos da AWS, como Amazon S3, Amazon EC2 e DynamoDB, consulte Exemplos de políticas baseadas em identidade do IAM.

Tópicos

- Permitir que um usuário liste grupos, usuários, políticas e outros itens da conta para fins de relatório
- Permitir que um usuário gerencie a associação de um grupo
- Permitir que um usuário gerencie usuários do IAM
- Permitir que usuários definam a política de senha da conta
- Permitir que usuários gerem e recuperem relatórios de credenciais do IAM
- Permitir todas as ações do IAM (acesso de administrador)

Permitir que um usuário liste grupos, usuários, políticas e outros itens da conta para fins de relatório

A política a seguir permite que o usuário chamem qualquer ação do IAM que comece com a string `Get` ou `List` e gere relatórios. Para visualizar a política de exemplo, consulte IAM: permite acesso somente leitura ao console do IAM.

Permitir que um usuário gerencie a associação de um grupo

A política a seguir permite que os usuários atualizem a associação do grupo chamado MarketingGroup. Para visualizar a política de exemplo, consulte IAM: permite gerenciar a associação de um grupo de forma programática e no console.

Permitir que um usuário gerencie usuários do IAM

A política a seguir permite que um usuário execute todas as tarefas associadas ao gerenciamento de usuários do IAM, mas não execute ações em outras entidades, como a criação de grupos ou políticas. As ações permitidas incluem:

- Criar o usuário (a ação `CreateUser`).
- Excluir o usuário. Esta tarefa requer permissões para executar todas as seguintes ações: `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup` e `DeleteUser`.
- Listar os usuários na conta e em grupos (as ações `GetUser`, `ListUsers` e `ListGroupsForUser`).
- Listar e remover políticas do usuário (as ações `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy`, `DeleteUserPolicy`)
- Renomear ou alterar o caminho para o usuário (a ação `UpdateUser`). O elemento `Resource` deve incluir um nome de recurso da Amazon (ARN) que abrange o caminho da fonte e o caminho de destino. Para obter mais informações sobre caminhos, consulte Nomes amigáveis e caminhos.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

Várias permissões incluídas na política anterior permitem que o usuário execute tarefas no Console de gerenciamento da AWS. Os usuários que executam tarefas relacionadas ao usuário apenas na AWS CLI, nos AWS SDKs ou na API de consulta HTTP do IAM podem não precisar de determinadas permissões. Por exemplo, se os usuários já conhecerem o nome de recurso da Amazon (ARN) das políticas a serem desanexadas de um usuário, eles não precisarão da permissão

`iam:ListAttachedUserPolicies`. A lista exata de permissões que um usuário requer depende das tarefas que o usuário deve executar enquanto gerencia outros usuários.

As seguintes permissões na política permitem acesso a tarefas do usuário por meio do Console de gerenciamento da AWS:

- `iam:GetAccount*`
- `iam:ListAccount*`

Permitir que usuários definam a política de senha da conta

Você pode conceder a alguns usuários permissões para obter e atualizar a política de senha da sua Conta da AWS. Para visualizar a política de exemplo, consulte IAM: permite configurar os requisitos de senha da conta de forma programática e no console.

Permitir que usuários gerem e recuperem relatórios de credenciais do IAM

Você pode conceder aos usuários permissão para gerar e baixar um relatório que liste todos os usuários na sua Conta da AWS. O relatório também lista o status de diversas credenciais de usuário, incluindo senhas, chaves de acesso, dispositivos MFA e certificados de assinatura. Para obter mais informações sobre relatórios de credenciais do , consulte Gerar relatórios de credenciais para sua Conta da AWS. Para visualizar a política de exemplo, consulte IAM: gerar e recuperar relatórios de credenciais do IAM.

Permitir todas as ações do IAM (acesso de administrador)

Você pode conceder a alguns usuários permissões administrativas para executar todas as ações no IAM, incluindo o gerenciamento de senhas, chaves de acesso, dispositivos com MFA e certificados de usuário. No exemplo a seguir a política concede estas permissões.

> **Warning**
>
> Quando você concede a um usuário acesso total ao IAM, não há limite de permissões que um usuário possa conceder a si mesmo e aos outros. O usuário pode criar novas entidades (usuários ou perfis) do IAM e conceder a essas entidades acesso total a todos os recursos na sua Conta da AWS. Ao conceder a um usuário acesso total ao IAM, você está efetivamente fornecendo a ele acesso total a todos os recursos na sua Conta da AWS. Isso

inclui acesso para excluir todos os recursos. Você deve conceder essas permissões apenas a administradores confiáveis e aplicar autenticação multifator (MFA) a esses administradores.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Delegação temporária do IAM

Visão geral

A delegação temporária acelera a integração e simplifica o gerenciamento de produtos da Amazon e de parceiros da AWS que se integram às suas contas da AWS. Em vez de configurar manualmente vários serviços da AWS, você pode delegar permissões temporárias e limitadas que permitem que o provedor do produto conclua as tarefas de configuração em seu nome em minutos por meio de fluxos de trabalho de implantação automatizados. Você mantém o controle administrativo com os requisitos de aprovação e os limites de permissões, enquanto as permissões do provedor do produto expiram automaticamente após a duração aprovada, sem a necessidade de limpeza manual. Se o produto exigir acesso persistente para operações contínuas, o provedor poderá usar a delegação temporária para criar um perfil do IAM com um limite de permissões que define as permissões máximas do perfil. Todas as atividades dos provedores de produtos são monitoradas pelo AWS CloudTrail para monitoramento de conformidade e segurança.

Para trabalhar com solicitações de delegação temporária, você pode iniciar uma solicitação e revisar as solicitações. Para mais informações sobre monitoramento, consulte notificações e geração de logs do CloudTrail.

> **Note**
>
> Solicitações de delegação temporária só podem ser criadas por produtos da Amazon e parceiros qualificados da AWS que concluíram o processo de integração do recurso. Os clientes analisam e aprovam essas solicitações, mas não podem criá-las diretamente. Se você for um parceiro da AWS que deseja integrar a delegação temporária do IAM em seu produto, consulte o Guia de integração de parceiros para obter instruções de integração e inclusão.

Como funciona a delegação temporária

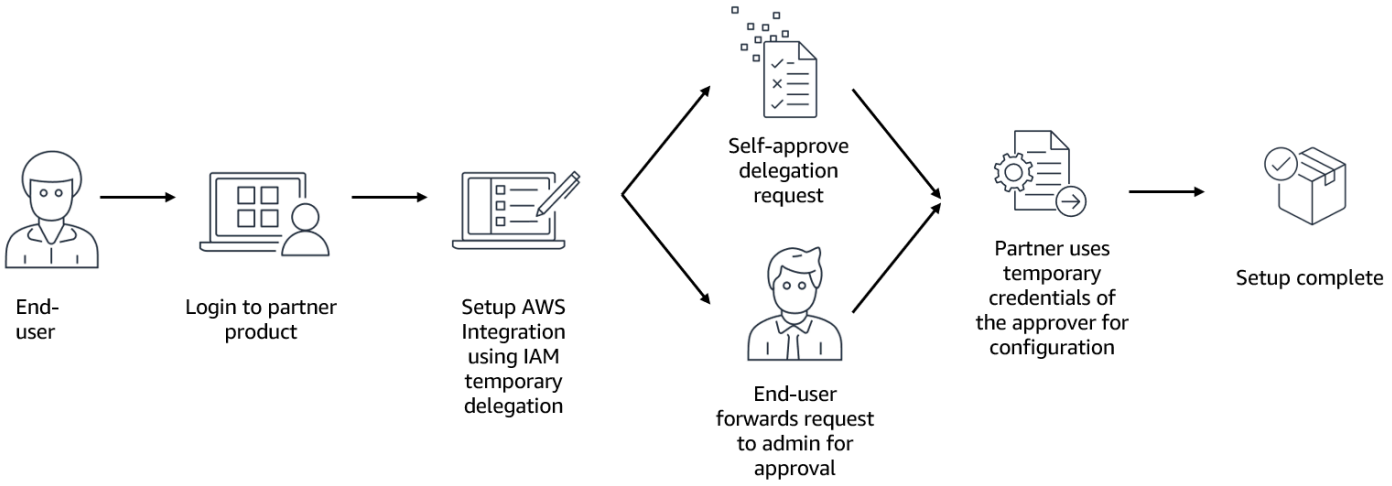
A delegação temporária permite que a Amazon e os parceiros da AWS solicitem acesso temporário e limitado à sua conta. Após sua aprovação, eles podem usar permissões delegadas para realizar ações em seu nome. As solicitações de delegação definem permissões específicas para os serviços da AWS e ações que o provedor do produto precisa para implantar ou configurar recursos em sua conta da AWS. Essas permissões só estão disponíveis por tempo limitado e expiram automaticamente após a duração especificada na solicitação.

> **Note**
>
> A duração máxima do acesso delegado é de 12 horas. No entanto, os usuários-raiz só podem aprovar solicitações de delegação com duração de quatro horas ou menos. Se uma solicitação especificar mais de quatro horas, você deverá usar uma identidade não raiz para aprovar a solicitação. Para obter detalhes, consulte Capacidade beta de simulação de permissão.

Para tarefas contínuas, como a leitura de um bucket do Amazon S3, as solicitações de delegação podem incluir a criação de um perfil do IAM que permite acesso contínuo a recursos e ações após a expiração do acesso temporário. Os provedores de produtos devem anexar um limite de permissões a qualquer perfil do IAM criado por meio de delegação temporária. Os limites de permissões limitam as permissões máximas de um perfil, mas não concedem permissões sozinhos. Você pode analisar o limite da permissões como parte da solicitação antes de aprová-la. Para obter detalhes, consulte Limites de permissões.

O processo funciona da seguinte maneira:

1. Você faz login em um produto da Amazon ou de um parceiro da AWS para integrá-lo ao seu ambiente da AWS.
2. O provedor do produto inicia uma solicitação de delegação em seu nome e o redireciona para o Console de Gerenciamento da AWS.
3. Você analisa as permissões solicitadas e determina se deve aprovar, negar ou encaminhar a solicitação ao administrador.
4. Depois que você ou seu administrador aprovar a solicitação, o provedor do produto poderá obter as credenciais temporárias do aprovador para realizar as tarefas necessárias.
5. O acesso do provedor do produto expira automaticamente após o período especificado. No entanto, qualquer perfil do IAM criado por meio da solicitação de delegação temporária persiste além desse período, permitindo que o provedor do produto continue acessando os recursos e as ações para tarefas de gerenciamento contínuas.



> **Note**
>
> Você só poderá delegar permissões a um provedor de produto se tiver permissões para os serviços e as ações incluídos na solicitação de delegação temporária. Se você não tiver acesso às ações e aos serviços solicitados, o provedor do produto não receberá essas permissões quando você aprovar a solicitação.

Se a verificação de permissão mostrar que é provável que seja bem-sucedida, você poderá aprovar a solicitação de delegação temporária e continuar com o fluxo de trabalho.

Se a verificação de permissão mostrar que você pode não ter permissões suficientes, encaminhe a solicitação ao administrador para aprovação. Recomendamos notificar seu administrador sobre essa solicitação usando seu método preferencial, como um e-mail ou um tíquete.

Depois que o administrador aprovar a solicitação, o que acontece a seguir depende da configuração do provedor do produto:

- Se o provedor do produto solicitar acesso imediato, ele receberá automaticamente permissões temporárias e a duração do acesso começará.
- Se o provedor do produto tiver solicitado a liberação pelo proprietário (destinatário inicial), você deverá retornar à solicitação para compartilhar explicitamente o acesso temporário à conta antes que a duração do acesso comece. Os provedores de produtos geralmente usam essa opção quando precisam de informações adicionais de você, como a seleção de recursos ou os detalhes de configuração, para concluir a tarefa necessária.

Iniciar uma solicitação de delegação temporária

Você pode iniciar uma solicitação de delegação temporária somente de produtos compatíveis da Amazon ou de parceiros da AWS. Durante um fluxo de trabalho que ofereça suporte à delegação temporária, será solicitado que você conceda ao provedor do produto permissões temporárias e limitadas para configurar os recursos necessários da AWS em sua conta. Essa abordagem automatizada fornece uma experiência mais otimizada, eliminando a necessidade de configurar esses recursos manualmente.

Você pode revisar os detalhes da solicitação de delegação, como os perfis, as políticas e os serviços da AWS específicos do IAM de que o provedor do produto precisa antes de conceder acesso. Você pode aprovar a solicitação, se tiver permissões suficientes, ou encaminhar a solicitação ao administrador da conta para aprovação. Todo o acesso do provedor do produto tem limite de tempo e pode ser monitorado e revogado conforme necessário.

Para iniciar uma solicitação de delegação temporária

1. Navegue até o console de um produto compatível da Amazon ou de parceiros da AWS que exija integração com sua conta da AWS.
2. Selecione Implantar com delegação temporária do IAM. Observe que o nome da opção pode variar entre os produtos compatíveis. Consulte a documentação do provedor do produto para obter detalhes.

> **Note**
>
> Se você ainda não tiver se conectado ao Console de Gerenciamento da AWS, uma nova janela será aberta na página de login da AWS. Recomendamos que você faça login na sua conta da AWS antes de iniciar a solicitação de delegação temporária no console do produto. Para obter mais informações sobre como fazer login com base no seu tipo de usuário e nos recursos da AWS que deseja acessar, consulte o Guia de início de sessão da AWS.

3. Revise os detalhes da solicitação para confirmar o nome do produto e a conta da AWS do provedor do produto. Você também pode revisar a identidade da AWS que o provedor do produto usará para executar ações em seu nome.
4. Revise os detalhes do acesso para ver as permissões que serão delegadas temporariamente ao aprovar essa solicitação.

   - A seção Resumo das permissões fornece uma visão geral de alto nível gerada por IA que pode ajudar você a entender quais categorias de serviços da AWS podem ser acessadas e quais tipos de ações podem ser executadas em cada serviço.
   - Escolha Visualizar JSON para revisar as permissões específicas que o provedor do produto precisa implantar em sua conta da AWS, incluindo o escopo de acesso e as limitações de recursos.
   - Se o provedor do produto criar um perfil do IAM como parte de uma solicitação de delegação temporária, um limite de permissões deverá ser anexado ao perfil. Esses perfis do IAM têm permissões que continuam a permitir acesso a recursos e ações após a expiração da duração do acesso solicitado. Escolha Visualizar detalhes para revisar o limite de permissões, que define as permissões máximas que o perfil pode ter. O provedor do produto aplicará políticas adicionais ao perfil durante a criação que definam suas permissões reais. Essas políticas podem parecer ser mais restritas ou mais amplas do que o limite, dependendo de como o provedor do produto as define. No entanto, o limite de permissões garante que as permissões efetivas do perfil nunca excedam o que você vê durante a aprovação da solicitação, independentemente das políticas associadas ao perfil. Para obter mais informações, consulteLimites de permissões.
5. Revise os resultados da simulação de permissão. O recurso de simulação de permissão avalia automaticamente as permissões de sua identidade em relação às incluídas na solicitação. Com base nessa análise, uma recomendação é exibida indicando se você deve aprovar a

solicitação com sua identidade atual ou encaminhar a solicitação para um administrador. Para obter detalhes, consulte Capacidade beta de simulação de permissão.

6. Na caixa de diálogo, selecione como deseja continuar.

   - Selecione Permitir acesso quando sua identidade tiver permissões suficientes para permitir que o provedor do produto realize procedimentos de integração em seu nome. Quando você seleciona essa opção, a duração do acesso do provedor do produto começa depois que você fornece acesso.
   - Selecione Solicitar aprovação se sua identidade não tiver permissões suficientes para permitir que o provedor do produto realize procedimentos de integração em seu nome. Em seguida, escolha Criar solicitação de aprovação. Quando você seleciona essa opção, é criado um link de solicitação de delegação temporária que você pode compartilhar com o administrador da conta. Seu administrador pode acessar o Console de Gerenciamento da AWS ou usar o link de acesso para revisar as solicitações de delegação temporária para aprovar a solicitação e compartilhar o acesso temporário com o solicitante.

> **Note**
>
> Conceder acesso ao provedor do produto requer duas ações: aceitar a solicitação de delegação (`AcceptDelegationRequest`) e liberar o token de troca (`SendDelegationToken`). O Console de Gerenciamento da AWS executa as duas etapas automaticamente quando você aprova uma solicitação. Se você usar a AWS CLI ou a API, deverá executar as duas etapas separadamente.

Capacidade beta de simulação de permissão

Ao receber uma solicitação de delegação temporária, você mesmo pode aprová-la ou encaminhá-la ao administrador da conta para aprovação. Você só poderá delegar permissões a um provedor de produto se tiver permissões para os serviços e as ações incluídos na solicitação de delegação temporária. Se você não tiver acesso aos serviços e ações solicitados, o provedor do produto não receberá essas permissões, mesmo que elas estejam incluídas na solicitação.

Por exemplo, uma solicitação de delegação temporária exige a capacidade de criar um bucket do Amazon S3, iniciar e interromper instâncias no Amazon EC2 e assumir um perfil do IAM. A identidade que aprova a solicitação pode iniciar e interromper instâncias no Amazon EC2 e assumir um perfil do IAM, mas não tem permissão para criar um bucket do Amazon S3. Quando essa

identidade aprova a solicitação, o provedor do produto não consegue criar um bucket do Amazon S3, mesmo que essas permissões tenham sido incluídas na solicitação de delegação temporária.

Como você só pode delegar permissões que você já possui, é fundamental avaliar se você tem as permissões solicitadas antes de aprovar. O recurso beta de simulação de permissões ajuda nessa avaliação comparando suas permissões com as incluídas na solicitação. A avaliação indica se você pode aprovar a solicitação com sua identidade atual ou se precisa encaminhá-la para um administrador. Se a análise não puder validar que você tem permissões suficientes, encaminhe a solicitação a um administrador para análise. Essa avaliação é baseada em uma análise simulada de permissões e pode ser diferente do seu ambiente ativo da AWS, portanto, analise as permissões solicitadas com cuidado antes de continuar.

Próximas etapas

Depois de iniciar uma solicitação de delegação temporária, você pode gerenciar e monitorar a solicitação ao longo de todo o seu ciclo de vida. Os seguintes procedimentos ajudam você a rastrear, aprovar e controlar o acesso temporário:

- Revisar as solicitações de delegação temporária: monitore o status de suas solicitações de acesso e visualize informações detalhadas sobre as solicitações para aprovar ou negar solicitações de delegação temporária.
- Revogar o acesso de delegação temporária: encerre imediatamente as sessões ativas de delegação temporária antes que elas expirem naturalmente.

Analisar as solicitações de delegação temporária

Depois de iniciar uma solicitação de delegação temporária, você pode monitorar, aprovar e rejeitar solicitações no console do IAM. A página de solicitações de delegação temporária fornece uma visão centralizada de todas as solicitações, incluindo aquelas que estão com a aprovação pendente, concluídas ou rejeitadas. Como administrador, você pode analisar essas solicitações para conceder aos provedores de produtos acesso aos recursos da AWS ou rejeitá-las com base nas políticas de segurança, nos requisitos do negócio ou nos padrões de conformidade da sua organização. Essa visibilidade ajuda você a monitorar o ciclo de vida do acesso do provedor do produto e a manter a supervisão das permissões temporárias.

> **Note**
>
> Você deve ter a permissão iam:AcceptDelegationRequest para aprovar solicitações de delegação temporária.

Para aprovar uma solicitação de delegação temporária

1. Faça login no Console de Gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Solicitações de delegação temporária.
3. A página principal exibe uma lista das solicitações de delegação temporária com as seguintes informações:
   - ID da solicitação: identificador exclusivo da solicitação
   - Status: status atual (pendente, aprovado, rejeitado, compartilhado, expirado)
   - Solicitante: provedor do produto associado à solicitação
   - Iniciado por: a entidade principal do IAM na conta que iniciou a solicitação para o provedor do produto
   - Solicitação criada: quando a solicitação foi enviada
   - Expiração da solicitação: quando a solicitação expirou ou expirará
4. (Opcional) Use as opções de filtro para visualizar as solicitações por status:
   - Todas as solicitações: visualize todas as suas solicitações, independentemente do status
   - Pendente: visualize as solicitações que aguardam a aprovação do administrador
   - Aprovada: visualize as solicitações aprovadas
   - Compartilhada: visualize as solicitações para as quais o acesso foi compartilhado
   - Rejeitada: visualize as solicitações rejeitadas com os motivos da rejeição
5. Para ver as informações detalhadas sobre uma solicitação específica ou para analisar uma solicitação com a aprovação pendente, escolha o ID da solicitação.
6. Analise as informações detalhadas da solicitação:
   - Informações do provedor do produto
   - Motivo e justificativa doa solicitação
   - Duração solicitada

- Permissões solicitadas da AWS

7. Se você for um administrador que está analisando uma solicitação pendente, escolha uma das seguintes opções:

   - Para aprovar a solicitação, escolha Aprovar. Na caixa de diálogo de aprovação, é possível visualizar os resultados da simulação da permissão. Para obter mais informações, consulte Capacidade beta de simulação de permissão. Depois de confirmar a duração do acesso e sua identidade da AWS, escolha Aprovar para conceder acesso. Se o provedor do produto solicitar acesso imediato, ele receberá automaticamente permissões temporárias e a duração do acesso começará. Caso contrário, notifique a pessoa que iniciou a solicitação para liberar o acesso ao provedor do produto.
   - Para rejeitar a solicitação, escolha Rejeitar.
     - Na caixa de diálogo de rejeição, forneça um motivo claro para a rejeição para ajudar o solicitante a entender por que sua solicitação foi negada.
     - Escolha Rejeitar para negar o acesso.

8. A lista de solicitações é atualizada automaticamente para mostrar as informações de status mais atuais. Você também pode atualizar a página manualmente para verificar se há atualizações de status.

Revogar o acesso temporário da delegação

Embora as sessões de acesso do provedor do produto tenham sido projetadas para expirar automaticamente após a duração aprovada, talvez seja necessário encerrar o acesso imediatamente em determinadas situações. A revogação do acesso ativo do provedor do produto fornece um mecanismo de controle de emergência quando surgem questões de segurança, quando o trabalho do provedor do produto é concluído antecipadamente ou quando os requisitos do negócio mudam. Tanto os iniciadores quanto os administradores da solicitação podem revogar o acesso para manter a segurança e o controle operacional.

Para revogar o acesso temporário da delegação

1. Faça login no Console de Gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Solicitações de delegação temporária.
3. Localize o ID da solicitação da sessão de acesso que você deseja revogar.

4. Escolha Ações e depois Revogar acesso.
5. Na caixa de diálogo, escolha Revogar acesso para confirmar que você deseja encerrar imediatamente a sessão de acesso.

Após revogar o acesso, o provedor do produto não poderá mais acessar seus recursos da AWS. A revogação é registrada em log no AWS CloudTrail para fins de auditoria.

> **Important**
>
> A revogação do acesso encerra imediatamente a sessão de acesso do provedor do produto. Qualquer trabalho ou processo em andamento usando o acesso será interrompido. Certifique-se de que a revogação não interrompa operações críticas.

> **Note**
>
> Você não pode revogar o acesso para solicitações que foram aprovadas usando um usuário-raiz. A AWS recomenda que você evite usar um usuário-raiz para aprovar solicitações de delegação. Use um perfil do IAM com as permissões adequadas.

Gerenciamento de permissões para solicitações de delegação

Os administradores podem conceder permissões às entidades principais do IAM para gerenciar solicitações de delegação de provedores de produtos. Isso é útil quando você deseja delegar autoridade de aprovação a usuários ou equipes específicos em sua organização ou quando precisa controlar quem pode realizar ações específicas em solicitações de delegação.

As seguintes permissões do IAM estão disponíveis para gerenciar solicitações de delegação:

Permissão	Descrição
iam:AssociateDelegationRequest	Associe uma solicitação de delegação não atribuída à sua conta da AWS
iam:GetDelegationRequest	Visualize detalhes de uma solicitação de delegação

Permissão	Descrição
iam:UpdateDelegationRequest	Encaminhe uma solicitação de delegação a um administrador para aprovação
iam:AcceptDelegationRequest	Aprove uma solicitação de delegação
iam:SendDelegationToken	Libere o token de troca para o provedor do produto após a aprovação
iam:RejectDelegationRequest	Rejeite uma solicitação de delegação
iam:ListDelegationRequests	Liste solicitações de delegação para sua conta

> **Note**
>
> Por padrão, as entidades principais do IAM que iniciam uma solicitação de delegação recebem automaticamente permissões para gerenciar essa solicitação específica. Elas podem associá-la à sua conta, visualizar os detalhes da solicitação, rejeitar uma solicitação, encaminhá-la a um administrador para aprovação, liberar o token de troca para o provedor do produto após a aprovação do administrador e listar as solicitações de delegação de sua propriedade.

Notificações

A delegação temporária do IAM se integra às Notificações de Usuários da AWS para ajudar você a se manter informado sobre as mudanças no estado da solicitação de delegação. As notificações são particularmente úteis para administradores que precisam revisar e aprovar solicitações de delegação.

Com as Notificações de Usuários da AWS, você pode configurar alertas para serem entregues por meio de vários canais, incluindo e-mail, o Amazon Simple Notification Service (SNS), o AWS Chatbot para Slack ou Microsoft Teams e o aplicativo móvel do Console da AWS. Isso garante que as pessoas certas sejam notificadas no momento certo, permitindo uma resposta mais rápida às aprovações pendentes ou o conhecimento das mudanças de acesso. Você também pode

personalizar quais eventos acionam notificações com base nas necessidades e nos requisitos de segurança da sua organização.

Eventos de notificação disponíveis

É possível se inscrever para receber notificações para os seguintes eventos de delegação temporária do IAM:

- Solicitação de delegação temporária do IAM criada
- Solicitação de delegação temporária do IAM atribuída
- Solicitação de delegação temporária do IAM com aprovação pendente
- Solicitação de delegação temporária do IAM rejeitada
- Solicitação de delegação temporária do IAM aceita
- Solicitação de delegação temporária do IAM finalizada
- Solicitação de delegação temporária do IAM expirada

Configuração de notificações

Para configurar notificações para eventos de delegação temporária do IAM:

1. Abra o console das Notificações de Usuários da AWS
2. Crie ou atualize uma configuração de notificação
3. Selecione AWS IAM como o serviço
4. Escolha sobre quais eventos de solicitação de delegação você deseja ser notificado
5. Configure seus canais de entrega (e-mail, AWS Chatbot etc.)

Para obter instruções detalhadas sobre como configurar as Notificações de Usuários da AWS, incluindo a configuração de canais de entrega e o gerenciamento de regras de notificação, consulte a documentação de Notificações de Usuários da AWS.

CloudTrail

Todas as ações realizadas pelos provedores de produtos usando acesso delegado temporário são registradas automaticamente no AWS CloudTrail. Isso fornece uma visibilidade e auditabilidade

completas da atividade do provedor do produto em sua conta da AWS. Você pode identificar quais ações foram executadas pelos provedores de produtos, quando elas ocorreram e qual conta de provedor de produto as executou.

Para ajudar você a distinguir entre as ações executadas por suas próprias entidades principais do IAM e aquelas realizadas por provedores de produtos com acesso delegado, os eventos do CloudTrail incluem um novo campo chamado `invokedByDelegate` abaixo do elemento `userIdentity`. Esse campo contém o ID da conta da AWS do provedor do produto, facilitando a filtragem e a auditoria de todas as ações delegadas.

Estrutura de eventos do CloudTrail

O exemplo abaixo mostra um evento do CloudTrail para uma ação realizada por um provedor de produtos usando acesso delegado temporário:

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "[REDACTED:AWS_ACCESS_KEY]",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedByDelegate": {
            "accountId": "444455556666"
        }
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
```

```
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePolicyName"
    },
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

O campo `invokedByDelegate` contém o ID da conta da AWS do provedor do produto que executou a ação usando o acesso delegado. Neste exemplo, a conta 444455556666 (o provedor do produto) executou uma ação na conta 111122223333 (a conta do cliente).

Delegação temporária do IAM para parceiros da AWS

Visão geral

A delegação temporária do IAM permite que os clientes da AWS incluam e/ou integrem de forma otimizada os produtos de parceiros da AWS em seu ambiente da AWS por meio de fluxos de trabalho interativos e guiados. Os clientes podem conceder aos parceiros da AWS acesso limitado e temporário para configurar os serviços necessários da AWS, reduzindo o atrito na integração e acelerando a geração de valor.

Para integrar a delegação temporária ao seu produto, aprenda a entender sua integração, analise as permissões de entendimento e siga as diretrizes de avaliação de políticas. Para exemplos de políticas do IAM, consulte modelos de políticas. Quando estiver tudo pronto para implementar, consulte como criar sua integração.

A delegação temporária do IAM permite que os parceiros:

- Simplifiquem a integração de clientes com o provisionamento automatizado de recursos
- Reduzam a complexidade da integração eliminando as etapas de configuração manual
- Criem confiança por meio de permissões transparentes e aprovadas pelo cliente
- Habilitem operações contínuas com padrões de acesso de longo prazo usando os limites de permissões

Como funciona

1. Parceiro cria uma solicitação de delegação: os parceiros criam uma solicitação especificando quais permissões precisam e por quanto tempo.
2. Avaliações de clientes no Console da AWS: o cliente vê exatamente quais permissões o parceiro está solicitando e o porquê.
3. Cliente aprova: o cliente aprova a solicitação e libera um token de troca. O token é enviado ao parceiro nesse tópico específico do SNS.
4. Parceiro recebe credenciais temporárias: os parceiros trocam o token por credenciais temporárias da AWS.
5. Parceiro configura os recursos: os parceiros usam as credenciais para configurar os recursos necessários na conta do cliente

Qualificação de parceiros

Para se qualificar para a integração de delegação temporária, um parceiro deve atender aos seguintes requisitos:

- Participação no ISV Accelerate: você deve estar inscrito no programa ISV Accelerate (ISVA).
- Listagem no AWS Marketplace: seu produto deve estar listado no AWS Marketplace com o selo “Implantado na AWS”.

Processo de integração

Conclua as seguintes etapas para integrar a delegação temporária ao seu produto:

1. Etapa 1: revisar os requisitos

   Analise esta documentação para entender os requisitos de qualificação e preencher o questionário de parceiro abaixo.
2. Etapa 2: enviar sua solicitação de integração

   Envie um e-mail para aws-iam-partner-onboarding@amazon.com ou entre em contato com seu representante da AWS. Inclua seu questionário de parceiro preenchido com todos os campos obrigatórios da tabela abaixo.
3. Etapa 3: validação e revisão da AWS

AWSO :

- Valide que você atende aos critérios de qualificação.
- Revise seus modelos de política e limites de permissões.
- Forneça feedback sobre os artefatos enviados.

4. Etapa 4: refinar suas políticas

   Responda ao feedback da AWS e envie modelos de políticas ou limites de permissão atualizados, conforme necessário.

5. Etapa 5: concluir o registro

   Após a aprovação, a AWS vai:

   - Habilitar o acesso à API para suas contas especificadas
   - Compartilhar ARNs para seu modelo de política e limite de permissões (se aplicável)

   Você receberá uma confirmação quando a integração for concluída. Em seguida, você pode acessar as APIs de delegação temporária, CreateDelegationRequest e GetDelegatedAccessToken, de suas contas registradas e começar a integrar fluxos de trabalho de solicitação de delegação em seu produto.

Questionário de parceiro

A tabela a seguir lista as informações necessárias para a integração do parceiro:

Informações	Descrição	Obrigatório
AccountID da Central de Parceiros	ID da conta da sua conta da AWS registrada no Central de Parceiros da AWS.	Sim
PartnerId	ID do parceiro fornecido pela Central de Parceiros da AWS.	Não

Informações	Descrição	Obrigatório
AWSID de produto do Marketplace	ID de produto para seu produto fornecido pelo Central de Parceiros da AWS.	Sim
AWSAccountIDs da	A lista de seus IDs de conta da AWS que você deseja usar para chamar APIs de delegação temporária. Ela deve incluir suas contas de produção e de não produção/teste.	Sim
Nome do parceiro	Esse nome é exibido para os clientes no Console de Gerenciamento da AWS quando eles analisam sua solicitação de delegação temporária.	Sim
E-mails de contato	Um ou endereços de e-mail para entrar em contato com você sobre sua integração.	Sim
Domínio do solicitante	Seu domínio (por exemplo, www.exemplo.com)	Sim
Descrição da integração	Breve descrição do caso de uso que você deseja abordar usando esse recurso. Você pode incluir links de referênci a para sua documentação ou outro material público.	Sim
Diagrama de arquitetura	Diagrama de arquitetura ilustrando seus casos de uso de integração.	Não
Modelo de política	Você deve registrar pelo menos um modelo de política para esse recurso. O modelo de política define as permissões temporárias que você deseja solicitar nas contas da AWS dos clientes. Para obter mais informaçõ es, consulte a seção Modelo de política.	Sim
Nome do modelo de política	Nome do modelo de política que você deseja registrar.	Sim

Informações	Descrição	Obrigatório
Limiar de permissões	Se você quiser criar perfis do IAM nas contas dos clientes usando permissões temporárias, você deve registrar um limite de permissões no IAM. Os limites de permissões serão anexados aos perfis do IAM que você criar para limitar o máximo de permissões no perfil do IAM. Você pode usar políticas gerenciadas pela AWS selecionadas como um limite de permissões ou registrar um novo limite de permissões personalizado (JSON). Para obter mais informações, consulte a seção Limites de permissões.	Não
Nome do limite de permissões	O nome do seu limite de permissões. O formato é arn:aws:iam::partner:policy/permission_boundary/<partner_domain>/<policy_name>_<date> O nome da política deve incluir a data de criação como sufixo. O nome não pode ser atualizado depois que o limite de permissões é criado. Se você estiver usando uma política existente gerenciada pela AWS, forneça então o ARN da política gerenciada.	Não
Descrição do limite de permissões	Descrição do limite para o limite de permissões. Essa descrição não pode ser atualizada depois que o limite de permissões é criado.	Não

Noções básicas sobre a integração

Depois de concluir o processo de integração, você pode criar sua integração com a delegação temporária do IAM. Uma integração completa normalmente envolve três categorias principais de trabalho:

1. Experiência do usuário e design do fluxo de trabalho

Crie uma experiência de frontend na aplicação de parceiros que oriente os clientes no fluxo de trabalho de delegação temporária. A aplicação do parceiro deve:

- Apresentar um fluxo claro de integração ou configuração em que os clientes possam conceder acesso temporário. Identifique essa ação de forma clara, como “Implantar com delegação temporária do IAM”.
- Redirecionar os clientes para o Console de Gerenciamento da AWS para revisar e aprovar a solicitação de delegação usando o link do console retornado pela API CreateDelegationRequest.
- Fornecer mensagens apropriadas sobre quais permissões estão sendo solicitadas e o porquê. Os clientes podem ver essa mensagem na página de detalhes da solicitação de delegação.
- Gerenciar o retorno do cliente à sua aplicação após concluir a aprovação na AWS.

Melhores práticas para solicitações de delegação temporária

Ao implementar a delegação temporária para sua aplicação de parceiro, siga as melhores práticas definidas abaixo para ajudar os clientes a verificar a autenticidade e a precisão da solicitação.

1. Inclua contexto identificável pelo usuário nas mensagens de solicitação

Você deve incluir o contexto específico do usuário na mensagem de solicitação de delegação. Essas informações ajudam os usuários a identificar as solicitações do fluxo de trabalho de parceiros existentes e a distinguir solicitações legítimas. Informações sugeridas na mensagem de solicitação:

- Nome de usuário ou identificador da conta do cliente em seu serviço
- Nome do espaço de trabalho, ID da assinatura ou identificador da organização
- Recurso específico sendo acessado (nome do cluster, nome do projeto, ambiente)
- Identificador exclusivo da transação ou solicitação gerado para essa tentativa de delegação

Exemplo de mensagem de solicitação:

```
Request from Partner A workspace "production-analytics"
Account: user@example.com
Workspace ID: 1234ABCD
Cluster: ml-training-cluster-01
Request ID: 1111-2222-3333-4444
```

2. Opcional: inclua o ID da conta da AWS ao iniciar a delegação

Se o ID da conta do cliente estiver disponível, inclua-o na solicitação de delegação. Essa etapa de verificação cria uma ligação adicional entre o token de delegação e a intenção do cliente.

3. Projete mensagens de solicitação para verificação de segurança

Estruture suas mensagens de solicitação para permitir que os clientes verifiquem com confiança a legitimidade antes de conceder acesso. Requisitos de experiência do usuário:

- Exiba a mensagem de solicitação com destaque na interface da aplicação antes de redirecionar para a AWS
- Use uma linguagem clara e descritiva que se conecte diretamente à ação atual do cliente
- Evite mensagens genéricas que possam enviadas em qualquer solicitação de delegação
- Inclua detalhes suficientes para que os clientes verifiquem se a solicitação corresponde ao fluxo de trabalho pretendido
- Exiba o ID da conta da AWS que receberá acesso delegado
- Forneça uma explicação clara de quais permissões sua aplicação solicitará

Inclua um aviso de confirmação em sua aplicação antes de redirecionar os clientes para a tela de consentimento da AWS:

```
## Redirecting to IAM Temporary Delegation
You are about to grant [Your Service Name] temporary access to your AWS account.

Before clicking "Allow" on the AWS consent screen:
• Verify the request details match your current action
• Confirm the AWS account ID matches your intended account
• Make sure you initiated this request from [Your Service Name]
```

4. Recomendações de vinculação de sessão

Enquanto o AWS IAM gerencia o fluxo de autorização principal, implemente essas práticas em sua aplicação para fortalecer a integridade da sessão:

- Gere identificadores de solicitação exclusivos e de uso único para cada tentativa de delegação
- Associe solicitações de delegação à sessão da aplicação ativa do cliente
- Implemente a expiração apropriada para a solicitação, dependendo do seu processo de delegação

- Valide se os retornos de chamada da delegação correspondem ao contexto da solicitação de origem
- Registre todas as iniciações e conclusões da solicitação de delegação para monitoramento de segurança

Ao seguir essas melhores práticas, você ajuda a criar uma experiência de delegação temporária mais segura.

2. Integração de APIs

Use as APIs de delegação temporária do IAM para enviar e gerenciar solicitações de delegação. Uma vez que suas contas da AWS estiverem registradas, você poderá acessar as seguintes APIs:

- CreateDelegationRequest do IAM: cria uma solicitação de delegação para a conta da AWS de um cliente. Essa API retorna um link do console para o qual você redireciona os clientes para revisar e aprovar a solicitação.
- GetDelegatedAccessToken do AWS STS: recupera as credenciais temporárias da AWS depois que um cliente aprova sua solicitação de delegação. Use essas credenciais para executar ações na conta do cliente.

Sua integração deve lidar com todo o ciclo de vida das solicitações de delegação, incluindo a criação de solicitações, o monitoramento de seu status e a recuperação de credenciais temporárias quando aprovadas.

3. Configuração e orquestração de recursos

Após obter as credenciais temporárias, orquestre os fluxos de trabalho necessários para configurar os recursos na conta da AWS do cliente. Isso pode incluir:

- Chamar APIs de serviço da AWS diretamente para criar e configurar recursos
- Implantar a infraestrutura usando modelos do AWS CloudFormation
- Criar perfis do IAM para acesso contínuo (requer o uso de limites de permissões) Como medida de defesa em profundidade, recomendamos que você use nomes de perfil dinâmicos, por exemplo, um nome com um sufixo gerado aleatoriamente. Essa abordagem torna o nome do perfil em cada conta de cliente único e imprevisível.

Sua lógica de orquestração deve ser idempotente e lidar com as falhas de forma resiliente, pois os clientes podem precisar repetir o processo ou modificar suas aprovações de delegação.

Análise das permissões

Como parte do processo de integração de recursos, você precisará registrar políticas no IAM que definam as permissões que você deseja solicitar nas contas dos clientes da AWS. O processo de registro fornece uma experiência mais consistente para os clientes e ajuda a evitar armadilhas comuns na criação de políticas.

Durante o registro, a AWS avalia suas políticas em relação a um conjunto de validações. Essas validações têm como objetivo padronizar a formatação e a estrutura das políticas e fornecer proteções básicas contra antipadrões conhecidos. As validações também reduzem o risco de escalonamentos de privilégios, acesso não intencional entre contas e amplo acesso a recursos de alto valor nas contas dos clientes.

Tipos de permissões

AWSA considerará duas categorias de permissões: temporárias e de longo prazo.

Permissões temporárias

As permissões temporárias limitam as permissões atribuídas a qualquer sessão temporária de acesso delegado. As permissões temporárias são descritas nos modelos de política que são aplicados à sessão delegada. Os modelos oferecem suporte aos parâmetros que você fornece ao criar uma solicitação de delegação. Esses valores de parâmetros são então vinculados à sessão. Atualmente, as permissões temporárias funcionam da mesma forma que as políticas de sessão disponíveis no AWS STS: as políticas limitam a capacidade do usuário subjacente, mas não concedem nenhum acesso adicional. Para obter mais informações, consulte a documentação do AWS STS nas políticas de sessão.

Permissões de longo prazo

As permissões de longo prazo limitam as permissões de qualquer perfil criado ou gerenciado por meio do acesso temporário. As permissões de longo prazo são implementadas como limites de permissões do IAM. Você pode enviar um ou mais limites de permissões para a AWS como parte da integração. Depois da aprovação, a AWS compartilhará com você um ARN de política que você pode referenciar em suas políticas.

Essas políticas de limite têm duas características marcantes. Primeiro, elas são imutáveis. Se quiser atualizar as permissões, você pode registrar um novo limite de permissões. Depois, você

pode anexar o novo limite de permissões aos perfis de seus clientes enviando uma nova solicitação de delegação. Segundo, as políticas não são modeladas. Como a mesma política de limite é compartilhada globalmente, ela não pode ser alterada de forma individualizada por cliente.

> **Important**
>
> Os limites de permissões têm um limite máximo de tamanho de 6.144 caracteres.

> **Note**
>
> Se você quiser atualizar um limite de permissões ou modelo de política, entre em contato com o IAM em aws-iam-partner-onboarding@amazon.com. Uma vez que o novo limite de permissões for registrado, você poderá enviar uma solicitação de delegação aos clientes para atualizar o perfil do IAM e anexar o limite de permissões recém-registrado. Consulte a seção Exemplos para obter mais detalhes.

Exemplo de caso de uso: workload de processamento de dados

Considere um provedor de produtos que executa uma workload de processamento de dados nas contas dos clientes. O provedor precisa configurar a infraestrutura durante a integração inicial, mas também requer acesso contínuo para operar a workload.

Permissões temporárias (para a configuração inicial):

- Criar instâncias do Amazon EC2, a VPC e grupos de segurança
- Criar um bucket do Amazon S3 para dados processados
- Criar um perfil do IAM para operações em andamento
- Anexar um limite de permissões ao perfil do IAM

Permissões de longo prazo (perfil do IAM com limite de permissões para operações em andamento):

- Iniciar e interromper as instâncias do Amazon EC2 para executar trabalhos de processamento
- Ler dados de entrada de um bucket do Amazon S3
- Gravar resultados processados no bucket do Amazon S3

As permissões temporárias são usadas uma vez durante a integração para configurar a infraestrutura. O perfil do IAM criado durante esse processo tem um limite de permissões que limita suas permissões máximas somente às operações necessárias para o gerenciamento contínuo da workload. Isso garante que, mesmo que as políticas do perfil sejam modificadas, ele não exceda as permissões definidas no limite.

Diretrizes da avaliação de políticas

AWSA avaliará suas políticas enviadas em relação a um conjunto de diretrizes. As mesmas diretrizes de avaliação se aplicam tanto aos modelos de política quanto aos limites de permissões, com pequenas diferenças observadas quando apropriado.

Para fins de avaliação, os serviços são separados em grupos distintos. A distinção mais importante é para serviços sensíveis à segurança, que gerenciam o acesso, as credenciais e as chaves. As políticas que concedem acesso a esses serviços precisam se concentrar estritamente no trabalho que está sendo feito. Os serviços sensíveis à segurança incluem: AWS Identity and Access Management (IAM), AWS Key Management Service (KMS), AWS Resource Access Manager (RAM), AWS IAM Identity Center, AWS Organizations e AWS Secrets Manager.

Uma distinção secundária são os serviços que podem acessar dados além dos limites da conta. As políticas desses serviços devem incluir proteções para evitar o acesso não intencional entre contas.

Validações comuns

Todas as instruções de políticas devem seguir estas diretrizes:

- Todas as instruções devem incluir os campos Effect, Action (ou NotAction), Resource e Condition, nessa ordem
- Todas as ações em uma única instrução devem ser listadas em ordem alfabética
- Todos os ARNs incluídos na política devem seguir a sintaxe definida na documentação pública dos serviços relevantes
- Os campos NotAction só podem ser usados em instruções Deny
- As ações nas instruções Allow devem incluir um código de serviço. Caracteres curinga genéricos (“*”) não são permitidos

Restrições de serviços sensíveis à segurança

As seguintes restrições se aplicam aos serviços sensíveis à segurança mencionados acima:

- As ações nas instruções Allow devem ser mais específicas do que [service]:*
- As ações nas instruções Allow para modelos de política de acesso temporário não devem conter curingas
- As ações sensíveis, como iam:PassRole ou iam:CreateServiceLinkedRole, exigem escopo adicional, como recursos específicos ou verificações condicionais. Entre essas ações estão:
  - Transferência de perfil do IAM
  - Ações de modificação do perfil do IAM
  - Ações de modificação da política do IAM
  - AWSOperações criptográficas ou de gravação do KMS
  - AWSOperações de gravação ou compartilhamento do RAM
  - AWSOperações do Secrets Manager para recuperar ou modificar segredos ou modificar políticas de recursos
- Outras ações podem usar um recurso curinga, como iam:ListUsers ou iam:GetPolicy
- As ações que gerenciam credenciais, como iam:CreateAccessKey, são bloqueadas

Restrições específicas do IAM

Para o IAM:

- Somente operações de gravação limitadas são permitidas para perfis e políticas do IAM. Você não pode solicitar permissões em outros recursos do IAM, como usuários, grupos e certificados.
- A anexação de políticas ou as ações de gerenciamento de políticas em linha são limitadas a perfis com um limite de permissões. Os limites de permissões devem ser fornecidos pelo parceiro ou em uma lista de políticas gerenciadas permitidas pela AWS. As políticas gerenciadas pela AWS poderão ser permitidas se não concederem permissões altamente privilegiadas ou administrativas. Por exemplo, políticas gerenciadas pela AWS para funções de trabalho específicas ou a política SecurityAudit podem ser aceitáveis. A AWS analisará cada política gerenciada pela AWS caso a caso durante o processo de integração.
- O gerenciamento de políticas só é permitido para políticas com um caminho específico do parceiro: arn:aws:iam::@{AccountId}:policy/partner_domain.com/[feature]*
- As tags só podem ser aplicadas durante a criação de recursos e somente para perfis e políticas
- As verificações de iam:PassRole devem corresponder a um nome específico ou prefixo de caminho

AWS STSRestrições específicas do

Para AWS STS:

- sts:AssumeRole deve ter como escopo um ARN de perfil específico, um prefixo ARN de perfil, ou deve estar limitado a um conjunto de contas ou a um ID da organização/unidade organizacional

Restrições do Centro de Identidade do IAM

Para o Centro de Identidade do AWS IAM, as seguintes ações estão bloqueadas:

- Todas as ações relacionadas ao gerenciamento de permissões (por exemplo, sso:AttachCustomerManagedPolicyReferenceToPermissionSet)
- Modificações de usuários, grupos e membros do AWS Identity Store
- Gerenciamento de tags

AWSRestrições do Organizations

Para o AWS Organizations, somente ações de leitura serão permitidas.

Validações adicionais específicas do serviço

- As ações que adquirem segredos ou credenciais, como glue:GetConnection ou redshift:GetClusterCredentials, devem ter condições que correspondam a ARNs completos, prefixos ARN ou tags
- Para o Amazon Redshift: redshift:GetClusterCredentials só é permitido em um nome de banco de dados específico, e redshift:GetClusterCredentialsWithIAM só é permitido em um nome de grupo de trabalho específico

> **Note**
>
> Ao gerenciar os recursos do IAM na conta, recomendamos usar um caminho que indique seu nome, por exemplo, arn:aws:iam::111122223333:role/partner.com/rolename. Isso ajudará a diferenciar os recursos associados à sua integração e facilitar a descoberta, a auditoria e a análise para os clientes.

Requisitos de acesso entre contas

As instruções que potencialmente permitem o acesso entre contas devem incluir pelo menos um dos itens abaixo:

- Uma condição que especifica a conta ou a organização do recurso (por exemplo, aws:ResourceOrgId que corresponde a um ou mais valores esperados)
- Um campo de recurso que inclui uma conta específica (por exemplo, arn:aws:sqs:*:111122223333:*)
- Um campo de recurso que inclui uma conta que não é curinga e um nome completo do recurso (por exemplo, arn:aws:s3:::full-bucket-name)

> **Note**
>
> O acesso entre contas é um recurso sensível que exige uma justificativa comercial clara. A AWS analisará cuidadosamente a necessidade de acesso entre contas durante o processo de integração.

Modelos de políticas

Os modelos de políticas são um novo constructo do IAM projetado para definir permissões temporárias que os parceiros solicitam nas contas dos clientes. Como as políticas regulares do IAM, elas definem permissões usando instruções com os elementos Effect, Action, Resource e Condition. A principal diferença é que os modelos de política incluem parâmetros (como @{bucketName}) que são substituídos por valores reais quando você cria uma solicitação de delegação.

Como funcionam os modelos de políticas

Como parte do processo de integração, você registra seus modelos de política com a AWS. A AWS atribui a cada modelo um ARN exclusivo ao qual você faz referência ao criar solicitações de delegação.

Ao criar uma solicitação de delegação, você especifica:

- O ARN do modelo da política
- Os valores de parâmetros a serem substituídos no modelo

AWSA combina o modelo com seus valores de parâmetros para gerar uma política padrão do IAM. Os clientes revisam essa política final renderizada ao aprovar sua solicitação de delegação, vendo exatamente quais permissões serão concedidas.

> **Note**
>
> A política renderizada final tem um limite de tamanho máximo de 2.048 caracteres.

Confira um exemplo simples que mostra como a substituição do modelo funciona.

Modelo de política:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
}
```

Parâmetros fornecidos na solicitação de delegação:

```
{
    "Name": "bucketName",
    "Values": ["customer-data-bucket"],
    "Type": "String"
}
```

Política final renderizada (o que os clientes veem):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::customer-data-bucket/*"
        }
    ]
}
```

Sintaxe do modelo

Os modelos de política usam dois recursos principais para fornecer flexibilidade: substituição de parâmetros e instruções condicionais. A substituição de parâmetros permite que você defina espaços reservados em seu modelo que são substituídos por valores reais ao criar uma solicitação de delegação. As instruções condicionais permitem que você inclua ou exclua instruções de políticas inteiras com base nos valores dos parâmetros.

Substituição e tipos de parâmetros

Use a sintaxe @{parameterName} para definir parâmetros em seu modelo de política. Ao criar uma solicitação de delegação, você deve especificar o tipo de cada parâmetro.

String

Um valor único que é substituído diretamente no modelo.

Modelo:

```
"Resource": "arn:aws:s3:::@{bucketName}/*"
```

Parâmetros:

```
{
    "Name": "bucketName",
    "Values": ["my-bucket"],
    "Type": "String"
}
```

Resultado renderizado:

```
"Resource": "arn:aws:s3:::my-bucket/*"
```

StringList

Vários valores que geram várias entradas de recursos. Quando um parâmetro StringList é usado em um ARN de recurso, ele se expande para criar entradas de recursos separadas para cada valor.

Modelo:

```
"Resource": "arn:aws:s3:::@{bucketNames}/*"
```

Parâmetros:

```
{
    "Name": "bucketNames",
    "Values": ["bucket-1", "bucket-2"],
    "Type": "StringList"
}
```

Resultado renderizado:

```
"Resource": [
    "arn:aws:s3:::bucket-1/*",
    "arn:aws:s3:::bucket-2/*"
]
```

Comportamento entre produtos

Quando vários parâmetros são usados no mesmo ARN de recurso, os parâmetros StringList criam um produto cruzado de todas as combinações.

Modelo:

```
"Resource": "arn:aws:s3:::@{bucketNames}/@{prefix}/*"
```

Parâmetros:

```
[
    {
        "Name": "bucketNames",
        "Values": ["bucket-1", "bucket-2"],
        "Type": "StringList"
    },
    {
```

```
        "Name": "prefix",
        "Values": ["data"],
        "Type": "String"
    }
]
```

Resultado renderizado:

```
"Resource": [
    "arn:aws:s3:::bucket-1/data/*",
    "arn:aws:s3:::bucket-2/data/*"
]
```

Instruções condicionais

Use a diretiva @Enabled para incluir ou excluir condicionalmente instruções inteiras com base nos valores dos parâmetros.

Sintaxe:

- @Enabled: “parameterName”: inclua a instrução quando o valor do parâmetro for “True”
- @Enabled: “!parameterName”: inclua a instrução quando o valor do parâmetro NÃO for “True” (negação)

Modelo:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "@Enabled": "ENABLE_S3_WRITE",
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::@{bucketName}/*"
        }
    ]
```

```
}
```

Parâmetros (quando ENABLE_S3_WRITE for “True”):

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["True"],
        "Type": "String"
    }
]
```

Resultado renderizado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": ["s3:PutObject"],
            "Resource": "arn:aws:s3:::my-bucket/*"
        }
    ]
}
```

Parâmetros (quando ENABLE_S3_WRITE for “False”):

```
[
    {
        "Name": "bucketName",
        "Values": ["my-bucket"],
        "Type": "String"
```

```
    },
    {
        "Name": "ENABLE_S3_WRITE",
        "Values": ["False"],
        "Type": "String"
    }
]
```

Resultado renderizado:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["s3:GetObject"],
            "Resource": "*"
        }
    ]
}
```

Quando ENABLE_S3_WRITE for definido como “True”, a instrução condicional será incluída. Quando for definido como “False”, a instrução será excluída da política renderizada.

Exemplos adicionais

Os exemplos a seguir demonstram padrões comuns para o uso de modelos de políticas na delegação temporária. Eles focam a criação de perfis do IAM com limites de permissões para acesso de longo prazo e mostram estratégias diferentes para definir o escopo de permissões para recursos específicos. Esses exemplos ilustram como equilibrar flexibilidade e segurança usando técnicas como prefixos ARN, marcação de recursos e atualizações de limites de permissões.

Exemplo 1: concessão de acesso de longo prazo a recursos específicos

O limite de permissões abaixo é enviado como “SQSAccessorBoundar” para “partner.com”:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
```

```
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

> **Note**
>
> Isso inclui uma condição de mesma conta para evitar a concessão de acesso a filas em outras contas com políticas de recursos abertos. Uma referência direta ao ID da conta do cliente não pode ser incluída porque o limite é compartilhado entre todos os clientes e não pode ser modelado.

Como esta é a primeira versão dessa política, seu ARN é arn:aws:iam::partner:policy/permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15

O modelo de política abaixo foi enviado para permissões de acesso temporárias:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:ListQueues"
            ],
            "Resource": "arn:aws:sqs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
```

```
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
                }
            }
        }
    ]
}
```

Exemplo 2: uso de prefixos ARN

O limite de permissões pode especificar um prefixo ARN do recurso para limitar o acesso:

```
"Resource": "arn:aws:sqs:*:@{AccountId}:PartnerPrefix*"
```

Isso limita o acesso somente aos recursos com esse prefixo, reduzindo o escopo dos recursos acessíveis.

Exemplo 3: uso de tags para controle de acesso a recursos

Você pode marcar recursos durante o acesso delegado temporário e contar com essas tags para controle de acesso de longo prazo.

Limite de permissões que permite acesso a recursos marcados:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:ResourceTag/ManagedByPartnerDotCom": "false"
        },
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

Modelo de política para marcar novas filas na criação:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:CreateQueue",
        "sqs:TagQueue"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "Null": {
            "aws:RequestTag/ManagedByPartnerDotCom": "false"
        }
    }
}
```

Modelo de política para marcar filas preexistentes e criar o perfil:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sqs:TagQueue"
            ],
            "Resource": "arn:aws:sqs:*:@{AccountId}:@{QueueName}",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "ManagedByPartnerDotCom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:PutRolePermissionsBoundary",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_15"
```

```
                }
            }
        }
    ]
}
```

Essa abordagem permite que os clientes confirmem explicitamente quais recursos específicos podem ser acessados no longo prazo.

Exemplo 4: atualização do limite de permissões

Para atualizar um limite de permissões, registre uma nova versão com um novo sufixo de data e solicite permissão para substituí-la.

Limite de permissões atualizado com permissão adicional:

```
{
    "Effect": "Allow",
    "Action": [
        "sqs:DeleteMessage",
        "sqs:PurgeQueue",
        "sqs:ReceiveMessage",
        "sqs:SendMessage"
    ],
    "Resource": "arn:aws:sqs:*:*:*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
    }
}
```

Como segunda versão, a política tem o ARN: arn:aws:iam::partner:policy/permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_20

Modelo de política para atualizar o limite de permissões no perfil existente:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "iam:PutRolePermissionsBoundary"
            ],
            "Resource": "arn:aws:iam::@{AccountId}:role/partner.com/SQSAccessor",
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": "arn:aws:iam::partner:policy/
permissions-boundary/partner.com/SQSAccessorBoundary_2025_01_20"
                }
            }
        }
    ]
}
```

Os clientes devem aprovar essa solicitação de delegação para atualizar o limite de permissões no perfil existente.

Criação da sua integração

Noções básicas sobre o ciclo de vida de solicitações

Antes de criar sua integração, é importante entender o fluxo das solicitações de delegação, desde a sua criação até a conclusão.

Estados das solicitações

Uma solicitação de delegação passa pelos seguintes estados:

Estado	Descrição
Não atribuída	Solicitação criada, mas ainda não associada a uma conta de cliente e a uma entidade principal do IAM. A solicitação pode ter sido criada sem a especificação de uma conta de destino ou com um ID da conta de destino, mas ainda não reivindicada pelo proprietário da conta.
Atribuída	Solicitação associada a uma conta de cliente e aguardando análise.
Aprovação pendente	O cliente encaminhou a solicitação a um administrador para aprovação.
Aceito	Solicitação aprovada pelo cliente, mas o token de troca ainda não foi liberado.

Estado	Descrição
Finalizada	Token de troca liberado para o provedor do produto. O período de delegação (validade do token de troca) começa quando a solicitação chega ao estado Finalizada.
Rejeitado	Solicitação rejeitada pelo cliente.
Expirada	A solicitação expirou devido à inatividade ou ao tempo limite.

Transições de estados

Fluxo normal (caminho de aprovação)

- Não atribuída → Atribuída: o cliente associa a solicitação à sua conta
- Atribuída → Aceita OU Atribuída → Aprovação pendente: o cliente aprova a solicitação diretamente OU encaminha ao administrador para análise
- Aprovação pendente → Aceita: o administrador aprova a solicitação
- Aceita → Finalizada: o cliente libera o token de troca

Caminho de rejeição

- Atribuída → Rejeitada: o cliente rejeita a solicitação
- Aprovação pendente → Rejeitada: o administrador rejeita a solicitação
- Aceita → Rejeitada: o cliente revoga a aprovação antes de liberar o token

Caminho da expiração

As solicitações expiram automaticamente quando nenhuma ação é executada dentro do prazo especificado:

- Não atribuída → Expirada (1 dia)
- Atribuída → Expirada (7 dias)
- Aprovação pendente → Expirada (7 dias)
- Aceita → Expirada (7 dias)
- Rejeitada → Expirada (7 dias)

- Finalizada → Expirada (7 dias)

Status terminais

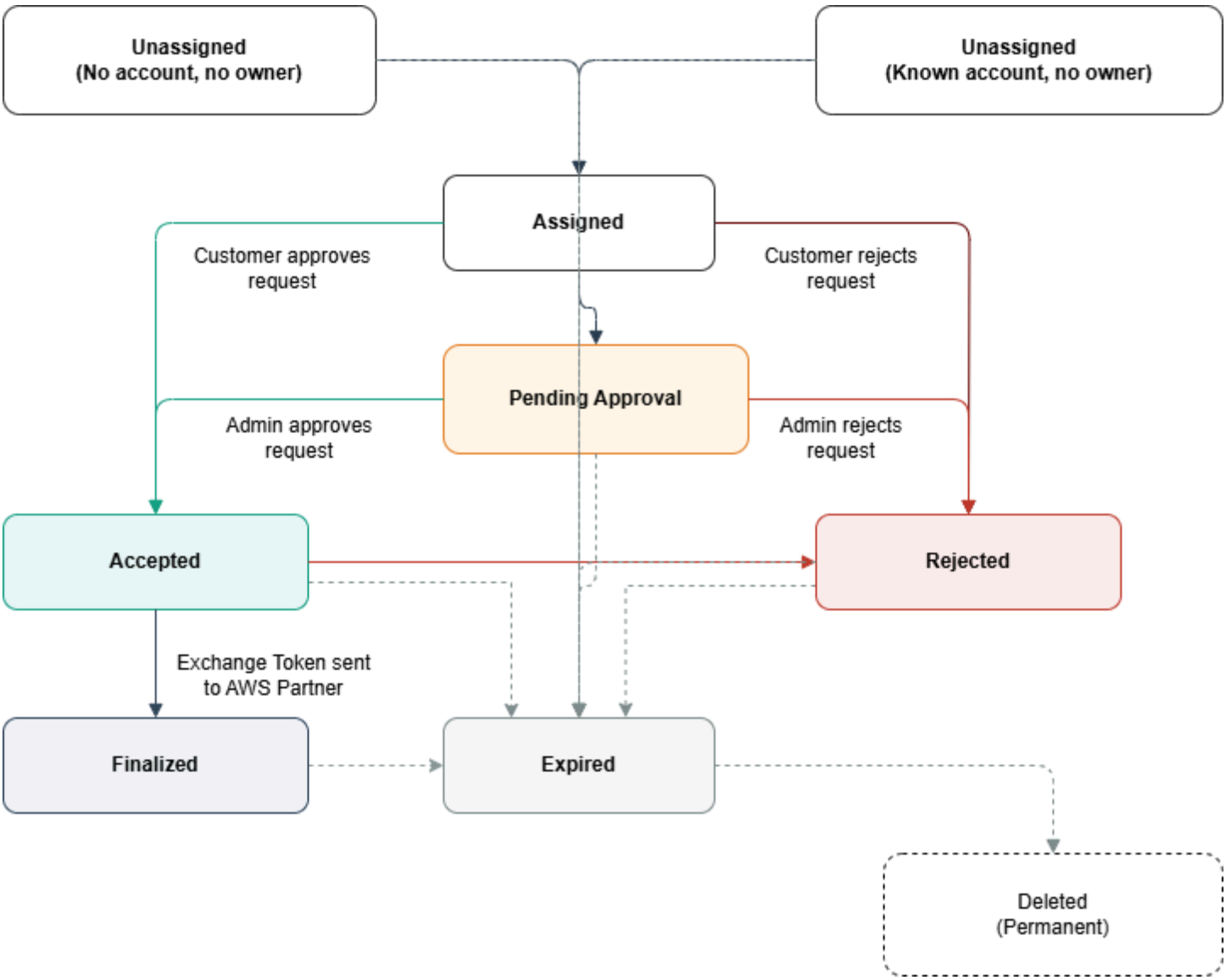
Os seguintes estados são terminais (sem mais transições):

- Finalizada: token de troca enviado
- Rejeitada: a solicitação foi negada
- Expirada: a solicitação atingiu o tempo limite ou o período de delegação terminou

As solicitações expiradas são excluídas do sistema após o período de retenção.

Gerenciamento dos estados de solicitação de delegação em sua aplicação

Como parceiro, você deve rastrear os estados de solicitação de delegação em seu sistema e apresentá-los aos seus clientes. Ao receber notificações do SNS sobre mudanças de estado, armazene essas atualizações em seu backend e as reflita na interface de usuário voltada para o cliente. Preste atenção especial ao estado Aprovação pendente: quando um cliente encaminha uma solicitação a um administrador para análise, a AWS envia uma notificação de Aprovação pendente para você. As solicitações podem permanecer nesse estado por até sete dias enquanto aguardam a ação do administrador. Durante esse período, mostre aos clientes que a solicitação deles está pendente e precisa da aprovação do administrador em sua aplicação. Considere fornecer um link direto para o Console da AWS, onde os clientes possam verificar o status da solicitação ou fazer o acompanhamento com o administrador. O gerenciamento correto da máquina de estado em seu backend e a apresentação das informações de estado corretas aos clientes em cada estágio é importante para uma boa experiência de integração.



Configuração de notificações

O IAM usa o Amazon Simple Notification Service (SNS) para comunicar a você as alterações de estado da solicitação de delegação. Ao criar uma solicitação de delegação, você deve fornecer um ARN do tópico do SNS da sua conta registrada da AWS. O IAM publicará mensagens sobre esse tópico para eventos importantes, incluindo quando os clientes aprovarem ou rejeitarem solicitações e quando o token de troca estiver pronto.

> **Note**
>
> Os tópicos do SNS não podem estar em regiões de aceitação da AWS. Seu tópico do SNS deve estar em uma região da AWS habilitada por padrão. Para obter uma lista de regiões de

aceitação, consulte Gerenciamento de regiões da AWS no Guia de gerenciamento de contas da AWS.

Configuração do tópico do SNS

Para receber notificações de solicitações de delegação, você deve configurar seu tópico do SNS para conceder permissões ao IAM para publicar mensagens nele. Adicione a seguinte instrução de política à política do tópico do SNS:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIAMServiceToPublish",
            "Effect": "Allow",
            "Principal": {
                "Service": "iam.amazonaws.com"
            },
            "Action": "SNS:Publish",
            "Resource": "arn:aws:sns:us-east-1:111122223333:TOPIC-NAME"
        }
    ]
}
```

Important

O tópico do SNS deve estar em uma de suas contas registradas da AWS. O IAM não aceitará tópicos do SNS de outras contas. Se a política do tópico não estiver configurada corretamente, você não receberá notificações de alteração de estado ou o token de troca.

Tipos de notificação

O IAM envia dois tipos de notificações:

Notificações de StateChange

Enviadas quando uma solicitação de delegação passa para um novo estado (Atribuída, Aprovação pendente, Aceita, Finalizada, Rejeitada, Expirada).

Notificações de ExchangeToken

Enviadas quando um cliente libera o token de delegação (estado Finalizada). Essa notificação inclui o token de troca de que você precisa para obter as credenciais.

Estados das notificações

É possível receber notificações para os seguintes estados de solicitações de delegação:

Estado	Tipo de notificação	Descrição
ATRIBUÍDA	StateChange	A solicitação foi associada a uma conta de cliente
APROVAÇÃO PENDENTE	StateChange	O cliente encaminhou a solicitação a um administrador para aprovação
ACEITA	StateChange	O cliente aprovou a solicitação, mas ainda não liberou o token
FINALIZADA	StateChange	O cliente liberou o token de troca
FINALIZADA	ExchangeToken	Essa notificação contém o token de troca
REJEITADA	StateChange	O cliente rejeitou a solicitação
EXPIRADA	StateChange	A solicitação expirou antes da conclusão

Formato das mensagens de notificação

O IAM publica notificações padrão do SNS. As informações da solicitação de delegação encontram-se no campo Mensagem como uma string JSON.

Campos comuns (todas as notificações)

Campo	Tipo	Descrição
Type	String	“StateChange” ou “ExchangeToken”
RequestId	String	O ID da solicitação de delegação do IAM

Campo	Tipo	Descrição
RequestorWorkflowId	String	O ID do fluxo de trabalho que você forneceu ao criar a solicitação
Estado	String	Estado atual da solicitação
OwnerAccountId	String	ID da conta da AWS do cliente
UpdatedAt	String	Carimbo de data e hora em que o estado mudou (formato ISO 8601)

Campos adicionais (somente notificações de ExchangeToken)

Campo	Tipo	Descrição
ExchangeToken	String	O token a ser trocado por credenciais usando a API GetDelegatedAccessToken do AWS STS
ExpiresAt	String	Quando o acesso delegado expira (formato ISO 8601)

Exemplo de notificações

Notificação de StateChange

```
{
  "Type": "Notification",
  "MessageId": "61ee8ad4-6eec-56b5-8f3d-eba57556aa13",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestorWorkflowId\":\"workflow-12345\",\"Type\":\"StateChange\",
\"RequestId\":\"dr-abc123\",\"State\":\"ACCEPTED\",\"OwnerAccountId\":\"111122223333\",
\"UpdatedAt\":\"2025-01-15T10:30:00.123Z\"}",
  "Timestamp": "2025-01-15T10:30:00.456Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```

Notificação de ExchangeToken


```
{
  "Type": "Notification",
  "MessageId": "e44e5435-c72c-5333-aba3-354406782f5b",
  "TopicArn": "arn:aws:sns:us-east-1:123456789012:partner-notifications",
  "Message": "{\"RequestId\":\"dr-abc123\",\"RequestorWorkflowId\":\"workflow-12345\",
\"State\":\"FINALIZED\",\"OwnerAccountId\":\"111122223333\",\"ExchangeToken\":
\"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...\",\"ExpiresAt\":\"2025-01-15T18:30:00.123Z\",
\"UpdatedAt\":\"2025-01-15T10:30:00.456Z\",\"Type\":\"ExchangeToken\"}",
  "Timestamp": "2025-01-15T10:30:00.789Z",
  "SignatureVersion": "1",
  "Signature": "...",
  "SigningCertURL": "...",
  "UnsubscribeURL": "..."
}
```


Tokens de troca

Um token de troca ou um token de intercâmbio é emitido pelo IAM quando um cliente aceita e finaliza uma solicitação de delegação. O provedor do produto usa esse token de troca ou de intercâmbio para chamar a API GetDelegatedAccessToken do AWS STS para obter credenciais temporárias da AWS com as permissões aprovadas pelos clientes. O token de troca em si não concede acesso aos seus recursos da AWS; ele deve ser trocado por credenciais reais por meio do AWS STS.

O token de troca só pode ser resgatado pela conta do provedor do produto que criou a solicitação de delegação. A conta solicitante está incorporada ao token, garantindo que somente o provedor autorizado do produto possa obter as credenciais para acessar a conta do cliente.

Duração do acesso

O período de delegação começa quando o cliente libera o token de troca, não quando o provedor do produto o resgata. Depois que o cliente liberar o token:

- O provedor do produto recebe o token via notificação do SNS
- Ele pode trocá-lo imediatamente por credenciais
- As credenciais expiram em: hora da liberação + duração aprovada
- O provedor do produto pode trocar o token várias vezes antes da expiração para obter novas credenciais, se necessário

Vários resgates

Os provedores de produtos podem trocar o token várias vezes durante o período de validade para obter novas credenciais. No entanto, todas as credenciais obtidas do mesmo token de troca expiram ao mesmo tempo, com base em quando você liberou o token.

Exemplo: se você aprovar uma solicitação de delegação de duas horas e liberar o token às 10h:

Hora de liberação do token	Tempo de troca do token	Expiração da credencial	Tempo utilizável
10:00 am	10:00 am	12:00h	2 horas
10:00 am	10h20	12:00h	1 hora e 40 minutos
10:00 am	11h40	12:00h	20 minutos
10:00 am	12:10h	Falha (o token expirou)	0 minuto

Conforme mostrado na tabela, trocar o token mais tarde no período de validade resulta em menos tempo utilizável para o provedor do produto.

Gerenciador de acesso à conta

O que é gerenciador de acesso à conta?

O gerenciador de acesso à conta permite que você atribua acesso à Conta da AWS aos usuários e grupos na instância da organização do IAM Identity Center. Você atribui acesso usando os perfis do IAM nas suas Contas da AWS. Você pode usar o gerenciador de acesso à conta junto com os conjuntos de permissões do IAM Identity Center ou sozinho. Tanto o gerenciador de acesso à conta quanto os conjuntos de permissões do IAM Identity Center usam os usuários e grupos que você sincronizou da sua fonte de identidade ou criou diretamente no IAM Identity Center.

Por que usar o gerenciador de acesso à conta?

O gerenciador de acesso à conta permite que você atribua os perfis do IAM existentes aos usuários e grupos que você sincronizou no IAM Identity Center com base na sua fonte corporativa de identidades. O uso dos perfis do IAM existentes oferece flexibilidade extra para personalizar o acesso do usuário, incluindo:

- Configure tags de perfil do IAM e use tags de sessão para o controle de acesso por atributo (ABAC).
- Configure a política de confiança do perfil do IAM para restringir a pressuposição de perfil ou usar as reivindicações declaradas pelo IdP como condições da política de confiança.
- Configure caminhos de perfil do IAM para agrupar os perfis de forma lógica e simplificar a administração. Para obter mais informações, consulte Configurar caminhos de perfil do IAM para agrupar perfis de forma lógica e simplificar a administração.

A tabela a seguir compara as três opções de acesso à conta:

	Perfis do IAM + gerenciador de acesso à conta	Conjuntos de permissões do IAM Identity Center	Perfis do IAM + federação do IAM (pressuposição direta do perfil)
Melhor para	Perfis do IAM personalizados que	Acesso humano basal uniforme,	Adequado para workloads

	Perfis do IAM + gerenciador de acesso à conta	Conjuntos de permissões do IAM Identity Center	Perfis do IAM + federação do IAM (pressuposição direta do perfil)
	variam entre as contas	como somente leitura e administr ador provisionado como perfis do IAM imutáveis	
Escopo	Várias contas (organização da AWS)	Várias contas (organização da AWS)	Por conta
Provisionamento de perfil do IAM	Requer infraestrutura como código (IaC) ou provisionamento manual por meio do console IAM da AWS	Automaticamente provisionado	Requer infraestrutura como código (IaC) ou provisionamento manual por meio dos consoles do IAM da AWS e do IdP externos
O mapeamento de perfil para identidade reside em	Gerenciador de acesso à conta	Centro de Identidade do IAM	IdP externo
Experiência do usuário final	Os usuários acessam as contas da AWS por meio do URL do portal de acesso à conta da AWS após fazer login na AWS como usuários do IAM Identity Center.	Os usuários acessam as contas da AWS por meio do portal de acesso da AWS após fazer login na AWS como usuários do IAM Identity Center.	Os usuários acessam as contas da AWS por meio de aplicativos SAML específicos da conta (ícones) em seu portal IdP.

	Perfis do IAM + gerenciador de acesso à conta	Conjuntos de permissões do IAM Identity Center	Perfis do IAM + federação do IAM (pressuposição direta do perfil)
Experiência do usuário da AWS CLI	1. O usuário faz login na AWS e acessa uma conta com um perfil específico em um navegador da web. 2. O usuário digita aws login na linha de comando para recuperar as credenciais da sessão do perfil do IAM do navegador . Durante a sessão de perfil do IAM, o usuário pode continuar trabalhando com a AWS CLI sem precisar se autenticar novamente na etapa 1.	O usuário configura um perfil, incluindo a conta desejada e o par de perfis, digitando aws configure sso. O usuário digita aws sso login para iniciar uma nova sessão usando um perfil. Se não houver uma sessão ativa do IAM Identity Center, o usuário se autentica em um navegador e autoriza o acesso nele. Para obter mais detalhes, consulte o Guia do usuário do Centro de Identidade do AWS IAM.	Integrações personalizadas usando provedores de credenciais do SDK da AWS

Capacidades gerais

Portal de acesso à conta

O gerenciador de acesso à conta fornece aos usuários um portal de acesso à conta, no qual eles podem visualizar e acessar as Contas da AWS atribuídas a eles por meio do gerenciador de acesso à conta.

Acesso à CLI

Sua força de trabalho pode acessar as Contas da AWS por meio da AWS CLI usando o comando aws login após fazer login na AWS em um navegador.

APIs de acesso à conta

O gerenciador de acesso à conta tem seu próprio namespace de API, `account-access`, o qual é separado da API do IAM. Você usa esse namespace na AWS CLI e nos SDKs.

Conceitos básicos do gerenciador de acesso à conta

Veja a seguir os conceitos básicos do gerenciador de acesso à conta. Isso pressupõe que os perfis do IAM que você deseja atribuir já existam nas respectivas Contas da AWS. Se esses perfis não existirem, consulte Criar um perfil usando políticas de confiança personalizadas para obter informações sobre como criar novos perfis.

Conceitos básicos do gerenciador de acesso à conta

1. Certifique-se de que você tenha uma instância de organização do IAM Identity Center habilitada e os usuários e grupos da sua força de trabalho provisionados nela. Para obter mais informações, consulte a seção Conceitos básicos do IAM Identity Center no Guia do usuário do Centro de Identidade do AWS IAM. Para obter mais detalhes, consulte the section called “Pré-requisitos e considerações” mais adiante neste tópico.

   > **Note**
   >
   > Se você estiver usando o IAM Identity Center replicado para várias Regiões da AWS ou com uma chave gerenciada pelo cliente (CMK), deve adicionar a permissão `kms:Decrypt` para que o gerenciador de acesso à conta use essa CMK ao se conectar ao IAM Identity Center. Para obter mais informações, consulte Política de chave basal do KMS no Guia do usuário do Centro de Identidade do AWS IAM.

2. Certifique-se de que você tenha permissões para habilitar o gerenciador de acesso à conta e gerenciar as atribuições de conta nele. Para obter mais informações, consulte the section called “Gerenciamento de identidade e acesso para o gerenciador de acesso à conta”.
3. Habilite o gerenciador de acesso à conta por meio do console ou da API.
4. Atualize seus firewalls e gateways para acessar o portal de acesso à conta.

5. Conceder ao gerenciador de acesso à conta acesso aos perfis do IAM atualizando sua política de confiança.
6. Comece a atribuir acesso à conta aos usuários e grupos da sua força de trabalho do IAM Identity Center.
7. Permita que os usuários saibam como acessar as Contas da AWS usando as atribuições de perfil do gerenciador de acesso à conta.

Pré-requisitos e considerações

Os seguintes pré-requisitos e considerações se aplicam:

- O gerenciador de acesso à conta está disponível nas Regiões comerciais da AWS habilitado por padrão. Regiões comerciais de adesão, GovCloud (EUA), China e outras regiões não comerciais não são compatíveis.
- Na habilitação, o gerenciador de acesso à conta se conecta a uma única instância de organização do IAM Identity Center. Antecipadamente, você deve habilitar uma organização da AWS e uma instância de organização do IAM Identity Center. As instâncias de conta do IAM Identity Center não são compatíveis.
- Recomendamos habilitar o gerenciador de acesso à conta na mesma Região da AWS em que você habilitou sua instância do IAM Identity Center. Se você ainda não fez isso, consulte Considerações para escolher uma Região da AWS no Guia do usuário do Centro de Identidade do AWS IAM.
- Tanto a instância de organização do IAM Identity Center quanto o gerenciador de acesso à conta devem estar habilitados na conta gerencial da sua organização.

Habilitar o gerenciador de acesso à conta

Depois de criar uma instância de organização do IAM Identity Center, você pode habilitar o gerenciador de acesso à conta por meio do console ou da API. Você deve fazer isso na mesma Conta da AWS e Região que a instância do IAM Identity Center.

Antes de habilitar o gerenciador de acesso à conta, execute o seguinte comando na conta gerencial da organização da AWS para conceder o acesso à sua organização para o gerenciador de acesso à conta:

```
aws organizations enable-aws-service-access --service-principal account-
access.amazonaws.com
```

> **Note**
>
> Embora o console do gerenciador de acesso à conta faça parte do console do IAM, a API reside em um namespace separado.

Use um dos métodos a seguir para habilitar o gerenciador de acesso à conta:

Console

Para habilitar o gerenciador de acesso à conta

1. Faça login na conta de gerenciamento de sua organização.
2. No painel de navegação do AWS Identity and Access Management console, escolha Gerenciador de acesso à conta.
3. Confirme se a Região do console está definida como a Região primária da sua instância do IAM Identity Center.
4. Se não houver instâncias de organização do IAM Identity Center, será solicitado que você habilite uma agora, antes de continuar. Caso contrário, escolha Habilitar gerenciador de acesso à conta.

AWS CLI

Primeiro, encontre o ARN da instância do IAM Identity Center e confirme se o código da Região corresponde à Região primária do IAM Identity Center. Em seguida, execute o seguinte comando:

```
aws account-access create-application \
  --region <Region> \
  --identity-source '{
    "identityCenter": {
      "instanceArn": "<IAM_IDENTITY_CENTER_INSTANCE_ARN>"
    }
  }'
```

Atualizar firewalls e gateways

O portal de acesso à conta fornece aos usuários acesso de logon único às Contas da AWS atribuídas a eles por meio do gerenciador de acesso à conta.

Se você filtrar o acesso a domínios ou endpoints de URL da AWS específicos usando uma solução de filtragem de conteúdo da Web, como firewalls de última geração (NGFW) ou Secure Web Gateways (SWG), deverá incluir na lista de permissões os domínios e endpoints de URL associados ao portal de acesso à conta.

A lista a seguir fornece os domínios de pilha dupla e os endpoints de URL de pilha dupla a serem adicionados às listas de permissões da solução de filtragem de conteúdo da web. O gerenciador de acesso à conta não tem endpoints somente para IPv4.

Lista de permissões de pilha dupla

URL do portal de acesso à conta:

- `https://[Tenant-ID].account-access.[Region].app.aws`

As APIs que o portal de acesso à conta chama:

- `[Tenant-ID].account-access.[Region].app.aws/api/*`
- `[Tenant-ID].account-access.[Region].app.aws/auth/*`

O ID do locatário é um identificador aleatório exclusivo associado ao gerenciador de acesso à conta. Para descobrir seu valor, navegue até a guia Configurações no console do gerenciador de acesso à conta e procure o URL do aplicativo na seção Detalhes da instância. O ID do locatário é o primeiro subdomínio no URL.

Os endpoints do serviço de gerenciador de acesso à conta:

- Ambiente de gerenciamento: `account-access.[Region].api.aws`
- Tempo de execução: `account-access-runtime.[Region].api.aws`
- Console: `console.aws.amazon.com/account-access`

Se você usar a AWS CLI, poderá encontrar o ID do locatário executando o seguinte comando da CLI na conta de administrador de gerenciamento ou delegado da sua organização.

```
aws account-access list-applications
```

A resposta contém informações sobre o gerenciador de acesso à conta, incluindo o ARN do aplicativo e o ID do locatário.

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

Considerações sobre a inclusão de domínios e endpoints de URL na lista de permissões

Além dos requisitos da lista de permissões para o portal de acesso da conta, os outros serviços e aplicações que você usa podem exigir a inclusão dos domínios na lista de permissões.

- Você precisa incluir o portal de acesso da AWS e respectivas dependências na lista de permissões. Para obter mais informações, consulte Atualizar firewalls e gateways para permitir o acesso ao portal de acesso da AWS no Guia do usuário do Centro de Identidade do AWS IAM.
- Para acessar as Contas da AWS, o Console de gerenciamento da AWS e o console do gerenciador de acesso à conta, você deve incluir domínios adicionais na lista de permissões. Consulte Troubleshooting no Console de gerenciamento da AWS Getting Started Guide para obter uma lista dos domínios do Console de gerenciamento da AWS.

Excluir gerenciador de acesso à conta

Quando o gerenciador de acesso à conta é excluído, todos os dados nele contidos são excluídos e não podem ser recuperados.

Use o procedimento a seguir para excluir o gerenciador de acesso à conta.

Console

Para excluir sua instância do gerenciador de acesso à conta

1. Faça login na conta de gerenciamento de sua organização.
2. No painel de navegação do console do AWS Identity and Access Management, escolha Gerenciador de acesso à conta.
3. Confirme se a Região do console está definida como a Região principal da instância do gerenciador de acesso à conta.
4. Escolha a guia Configurações.
5. Na seção Excluir gerenciador de acesso à conta, escolha Excluir.
6. A caixa de diálogo solicita a confirmação. Quando estiver tudo pronto, escolha Excluir.

AWS CLI

Execute o seguinte comando para excluir o gerenciador de acesso à conta:

```
aws account-access delete-application \
  --region "<Region>" \
  --application-arn "<account_access_manager_ARN>"
```

Warning

A exclusão do gerenciador de acesso à conta é irreversível e tem as seguintes consequências:

- Todas as atribuições de perfil são removidas permanentemente. Seus usuários perdem imediatamente a capacidade de acessar as Contas da AWS com perfis do IAM atribuídos por meio do gerenciador de acesso à conta.
- O gerenciador de acesso à conta é removido do portal de acesso da AWS para todos os usuários.
- As sessões ativas dos perfis do IAM atribuídos pelo gerenciador de acesso à conta continuam até expirarem, mas os usuários não podem iniciar novas sessões.
- Se você reativar o gerenciador de acesso à conta posteriormente, deverá recriar todas as atribuições do zero. As atribuições anteriores não podem ser recuperadas.

Antes de excluir, pense em desabilitar o acesso do usuário em contas individuais para restringir temporariamente o acesso.

Gerenciar o acesso ao Contas da AWS

O gerenciador de acesso à conta é integrado ao IAM Identity Center e ao AWS Organizations para gerenciar centralmente a atribuição de perfis do IAM existentes em várias Contas da AWS sem configurar cada uma de suas contas individualmente. Com o gerenciador de acesso à conta, você atribui os perfis do IAM existentes em suas Contas da AWS aos usuários e grupos do IAM Identity Center para controlar o acesso a Contas da AWS específicas.

Com o gerenciador de acesso à conta, suas equipes podem criar os próprios perfis do IAM personalizados nas respectivasContas da AWS, e você, como administrador, usa o gerenciador de acesso à conta para gerenciar centralmente as atribuições de usuários e grupos do IAM Identity Center para esses perfis do IAM.

Note

No contexto do gerenciador de acesso à conta, os termos usuários e grupos se referem exclusivamente aos usuários e grupos da força de trabalho no IAM Identity Center.

Administradores delegados

Quando você habilita o gerenciador de acesso à conta, a AWS o cria na conta gerencial no AWS Organizations porque o gerenciador de acesso à conta gerencia as atribuições do perfil do IAM para uma organização inteira da AWS. Você pode optar por delegar a administração do gerenciador de acesso à conta a uma conta de membro em AWS Organizations.

Se você optar por registrar uma conta de membro como administrador delegado do gerenciador de acesso à conta, os usuários dessa conta poderão fazer a maioria das tarefas administrativas do gerenciador de acesso à conta. A tabela a seguir indica se uma tarefa administrativa do gerenciador de acesso à conta pode ser executada por usuários em uma conta gerencial da organização, em uma conta de membro de administração delegada ou em outra conta de membro. O gerenciamento do perfil do IAM é excluído da lista de tarefas administrativas porque ocorre fora do gerenciador de acesso à conta.

Tarefas administrativas do gerenciador de acesso à conta	conta gerencial	Conta de administrador delegado	Conta de membro
Gerenciar atribuições de perfil do IAM*	Sim	Sim	Não
Habilitar gerenciador de acesso à conta	Sim	Não	Não
Excluir gerenciador de acesso à conta	Sim	Não	Não
Ative ou desative o acesso do usuário na conta de gerenciamento	Sim	Não	Não
Habilitar ou desabilitar o acesso do usuário em uma conta de membro	Sim	Sim	Não
Registre ou cancele o registro de uma conta-membro como administrador delegado	Sim	Não	Não

*Consulte as melhores práticas para administração delegada em relação às atribuições de usuários e grupos à conta gerencial.

Registre uma conta-membro

Para configurar a administração delegada, você deve primeiro registrar uma conta de membro em sua organização como administrador delegado. Os usuários da conta de membro que tiverem permissões suficientes terão acesso administrativo ao gerenciador de acesso à conta.

O gerenciador de acesso à conta suporta registrar apenas uma conta de membro como administrador delegado por vez. Você só pode registrar uma conta de membro enquanto estiver conectado com as credenciais da conta de gerenciamento.

Use o procedimento a seguir para registrar uma conta de membro em sua organização da AWS como administrador delegado.

Como registrar uma conta-membro

1. Faça login no Console de gerenciamento da AWS usando as credenciais da sua conta de gerenciamento da AWS Organizations. As credenciais da conta de gerenciamento são necessárias para executar a API RegisterDelegatedAdministrator.
2. Selecione a Região em que o gerenciador de acesso à conta foi habilitado e, em seguida, abra o console AWS Identity and Access Management.
3. Escolha Gerenciador de acesso à conta e, em seguida, escolha a guia Configurações.
4. Na seção Administrador delegado, escolha Registrar.
5. Na página Registrar administrador delegado, selecione a Conta da AWS que você deseja registrar e, em seguida, escolha Registrar.

Cancelar o registro de uma conta-membro

Você só pode cancelar o registro de uma conta de membro enquanto estiver conectado com as credenciais da conta de gerenciamento.

Use o procedimento a seguir para cancelar o registro de uma conta de membro em sua organização da AWS que já havia sido designada como administrador delegado.

> **Important**
>
> Ao cancelar o registro de uma conta, você remove a capacidade de todos os usuários administradores de operar o gerenciador de acesso à conta dessa conta. Eles não podem mais administrar o acesso à Conta da AWS dessa conta.
> Essa operação não afeta as permissões ou atribuições configuradas no gerenciador de acesso à conta. Seus usuários finais continuam acessando as Contas da AWS pelo portal de acesso à conta.

Como cancelar o registro de uma conta-membro

1. Faça login no Console de gerenciamento da AWS usando as credenciais de gerenciamento da sua conta em AWS Organizations As credenciais da conta de gerenciamento são necessárias para executar a API DeregisterDelegatedAdministrator.
2. Selecione a Região em que o gerenciador de acesso à conta foi habilitado e, em seguida, abra o console AWS Identity and Access Management.
3. Escolha Gerenciador de acesso à conta e, em seguida, a guia Configurações.
4. Na seção Administrador delegado, selecione Cancelar conta.
5. Na caixa de diálogo Cancelar registro do administrador delegado, analise as implicações de segurança e confirme que você entende.
6. Escolha Cancelar registro.

Exibir a conta de administrador delegado registrada

Use o procedimento a seguir para descobrir qual conta de membro no seu AWS Organizations foi configurada como administrador delegado do gerenciador de acesso à conta.

Como visualizar sua conta-membro registrada

1. Faça login no Console de gerenciamento da AWS usando as credenciais de gerenciamento da sua conta em AWS Organizations
2. Selecione a Região em que o gerenciador de acesso à conta foi habilitado e, em seguida, abra o console AWS Identity and Access Management.
3. Escolha Gerenciador de acesso à conta e, em seguida, a guia Configurações.
4. Na seção Administrador delegado, você pode ver os detalhes da conta atual do administrador delegado.

Preparar os perfis do IAM

Antes de começar a atribuir perfis do IAM aos usuários e grupos do IAM Identity Center, você precisa:

1. Identificar os perfis do IAM relevantes nas contas de interesse e
2. Certifique-se de que a política de confiança dos perfis do IAM contenha a instrução fornecida abaixo.

Se o perfil tiver uma política de confiança existente, adicione essa instrução à matriz de Instruções. Se você estiver criando um novo perfil, poderá usar o documento de política inteiro. Essa instrução de política permite que o gerenciador de acesso à conta assuma o perfil e defina o contexto em nome dos usuários. O bloco `Condition` fornece proteção contra confused deputy ao garantir que o gerenciador de acesso à conta possa assumir o perfil somente em nome da própria conta e do aplicativo de gerenciador de acesso à conta. Para obter mais informações, consulte Atualizar a política de confiança de perfis.

Substitua `<your-account-id>` pelo seu ID da Conta da AWS, `<region>` pela Região principal do IAM Identity Center e `<application-id>` pelo ID do aplicativo do gerenciador de acesso à conta. Você pode encontrar o ARN do aplicativo na página Configurações no console do gerenciador de acesso à conta.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

Opcional — **`sts:TagSession`**. Adicione `sts:TagSession` à lista de `Action` se quiser que os atributos de declaração a montante do seu provedor de identidade (por exemplo, associações a grupos, departamento, centro de custos) se propaguem como tags de entidade principal na sessão emitida pelo gerenciador de acesso à conta. Sem essa ação, a pressuposição do perfil ainda é bem-

sucedida; somente a propagação da tag de sessão é suprimida. Adicione quando as políticas do IAM a jusante na conta de destino usarem tags de entidade principal (`aws:PrincipalTag/<key>`); caso contrário, deixe de fora. Mantenha o mesmo bloco `Condition`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountAccessManagerIAMRoleTrustPolicyStatement",
      "Effect": "Allow",
      "Principal": {
        "Service": "account-access.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "<your-account-id>",
          "aws:SourceArn": "arn:aws:account-access:<region>:<your-account-
id>:application/<application-id>"
        }
      }
    }
  ]
}
```

> **Note**
>
> Recomendamos que você atualize seus fluxos de trabalho de criação de perfis do IAM para adicionar essa instrução de política de confiança automaticamente quando necessário.

Atribuir e remover acesso

Atribuir acesso de um usuário ou grupo a uma Conta da AWS

Você pode atribuir acesso a um usuário ou grupo a uma Conta da AWS criando uma atribuição entre o usuário ou grupo, a conta e o perfil do IAM que o usuário ou grupo pode assumir na conta. Assim,

uma única atribuição é um trio que consiste no ID do usuário ou grupo, no ID da conta e no ARN do perfil do IAM.

Se você quiser realizar atribuições em massa, como atribuir o mesmo perfil a vários usuários e grupos, deverá criar cada atribuição individualmente. Você pode criar um script para esse procedimento usando a API de acesso à conta e o SDK/CLI da AWS.

Console

O console fornece algumas maneiras diferentes de atribuir novos acessos, dependendo se você deseja primeiro identificar a conta, o usuário ou o grupo.

> **Note**
>
> O console do gerenciador de acesso à conta fornece um recurso de pesquisa para usuários, grupos e contas. Para identificar o perfil que você deseja atribuir, você deve encontrar o respectivo nome (ou ARN) com antecedência. O console do gerenciador de acesso à conta não consegue detectar perfis do IAM em Contas da AWS.

Em todos os procedimentos desta seção, faça login na conta gerencial (ou conta de administrador delegado), selecione a Região em que o gerenciador de acesso à conta está habilitado e navegue até Gerenciador de acesso à conta no console do IAM.

Accounts tab

Para atribuir acesso a usuários ou grupos na guia Contas

1. Selecione a guia Accounts.
2. Se você já sabe o ID da conta, escolha Atribuir novo acesso. Na página Atribuir acesso à conta, procure o usuário ou grupo desejado, insira o ID da conta e o nome do perfil do IAM e escolha Atribuir acesso.
3. Se precisar primeiro encontrar a conta, você pode pesquisar a Conta da AWS desejada pelo nome, ID ou endereço de e-mail. Como alternativa, você pode revisar a estrutura organizacional ou a lista de contas para encontrar as Conta da AWS relevantes. Escolha o botão de opção na frente da conta e, em seguida, Atribuir novo acesso. Na página Atribuir acesso à conta, procure o usuário ou grupo desejado, insira o nome do perfil do IAM e escolha Atribuir acesso.

4. A estrutura organizacional ou a lista de contas permite que você escolha uma conta para ver os respectivos detalhes e as atribuições de acesso à conta. Você também pode escolher Atribuir novo acesso aqui.

Users tab

Para atribuir acesso ao usuário na guia Usuários

1. Escolha a guia Users.
2. Você pode pesquisar usuários por nome de usuário e nome de exibição ou encontrá-los na lista de usuários exibida.
3. Escolha o usuário ao qual deseja atribuir o novo acesso.
4. A próxima página mostra os atributos do usuário, as atribuições da conta e as associações a grupos.
5. Escolha Atribuir novo acesso.
6. Na página Atribuir acesso à conta, insira o ID da conta e o nome do perfil do IAM e escolha Atribuir acesso.

Groups tab

Para atribuir acesso ao grupo na guia Grupos

1. No painel de navegação, escolha Gerenciador de acesso à conta.
2. Selecione a guia Grupos.
3. Você pode pesquisar grupos pelo nome do grupo ou encontrá-los na lista de grupos exibida.
4. Escolha o grupo ao qual deseja atribuir o novo acesso.
5. A próxima página mostra os atributos do grupo, as atribuições da conta e os membros usuários.
6. Escolha Atribuir novo acesso.
7. Na página Atribuir acesso à conta, insira o ID da conta e o nome do perfil do IAM e escolha Atribuir acesso.

AWS CLI

O seguinte comando da AWS CLI pode ser usado para criar uma atribuição de conta para um usuário:

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "userId": "<Identity_Center_User_ID>"
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
    }
  }'
```

O seguinte comando da AWS CLI pode ser usado para criar uma atribuição de conta para um grupo:

```
aws account-access create-entitlement \
  --region <Region> \
  --application-arn "<Account_access_manager_ARN>" \
  --entitlement '{
    "principalRole": {
      "principal": {
        "identityCenter": {
          "groupId": "<IDC_GROUP_ID>"
        }
      },
      "roleArn": "arn:aws:iam::<ACCOUNT_ID>:role/<ROLE_NAME>"
    }
  }'
```

Remover acesso dos usuários ou grupos a uma Conta da AWS

Existem várias maneiras para encontrar a atribuição a ser removida.

Console

Em todos os procedimentos desta seção, faça login na conta gerencial (ou conta de administrador delegado), selecione a Região em que o gerenciador de acesso à conta está habilitado e navegue até Gerenciador de acesso à conta no console do IAM.

Accounts tab

Para remover o acesso de usuários ou grupos na guia Contas

1. Selecione a guia Accounts.
2. Encontre a conta relevante e escolha o respectivo nome.
3. Na seção Atribuições de acesso à conta, escolha o usuário ou grupo do qual você deseja remover o acesso e escolha Cancelar atribuição.
4. No modal, escolha Cancelar atribuição para confirmar.

Users tab

Para remover o acesso do usuário na guia Usuários

1. Escolha a guia Users.
2. Escolha o usuário do qual deseja remover acesso.
3. Na seção Atribuições de acesso à conta, escolha a conta da qual você deseja remover o acesso e escolha Cancelar atribuição.
4. No modal, escolha Cancelar atribuição para confirmar.

AWS CLI

Para remover o acesso com a CLI, primeiro você precisa encontrar a atribuição da conta (direito) e, em seguida, usar seu ID no comando a seguir. Mais adiante neste tópico, veja como pesquisar atribuições de conta para uma conta, usuário, grupo ou perfil específico. A CLI se refere às atribuições como direitos.

```
aws account-access delete-entitlement \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --entitlement-id "<ENTITLEMENT_ID>"
```

Exibir ou pesquisar atribuições da conta

É possível visualizar atribuições da conta de várias maneiras.

Console

Em todos os procedimentos desta seção, faça login na conta gerencial (ou conta de administrador delegado), selecione a Região em que o gerenciador de acesso à conta está habilitado e navegue até Gerenciador de acesso à conta no console do IAM.

Accounts tab

Exibir o acesso do usuário ou grupo na guia Contas

1. Selecione a guia Accounts.
2. Por padrão, a visualização Hierarquia é exibida. Você pode alternar para a visualização Lista, se desejar. É possível pesquisar contas por nome, ID, e-mail e ID de UO. Quando encontrar a conta desejada, escolha o respectivo nome.
3. Na página da conta, você pode ver as atribuições da conta na seção Atribuições de acesso à conta. Você pode filtrá-las adicionalmente por nome de usuário, ID de usuário, nome do grupo, ID do grupo e ARN do perfil do IAM.

Users tab

Exibir o acesso do usuário na guia Usuários

1. Escolha a guia Users.
2. Pesquise o usuário por nome ou ID de usuário. Ao encontrar o usuário desejado, escolha o nome de usuário.
3. Na página do usuário, você pode ver as atribuições de conta do usuário. Você pode filtrá-las adicionalmente por ID da conta, nome da conta e ARN do perfil do IAM.

Groups tab

Exibir o acesso do grupo na guia Grupos

1. Selecione a guia Grupos.

2. Pesquise o grupo por nome ou ID do grupo. Ao encontrar o grupo desejado, escolha o nome do grupo.
3. Na página do grupo, você pode ver as atribuições de conta do grupo. Você pode filtrá-las adicionalmente por ID da conta, nome da conta e ARN do perfil do IAM.

AWS CLI

Os comandos a seguir ilustram como recuperar atribuições de conta usando filtros diferentes. A CLI se refere às atribuições como direitos.

Recuperar as primeiras 100 atribuições (direitos) de uma conta específica

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principalRole": {
        "account": "<ACCOUNT_ID>"
      }
    }' \
    --max-results 100
```

Recuperar todas as atribuições (direitos) de um usuário específico

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
      "principal": {
        "type": "USER",
        "id": "<USER_ID>"
      }
    }'
```

Recuperar todas as atribuições (direitos) de um grupo específico

```
aws account-access list-entitlements \
    --region <Region> \
    --application-arn "<Account_access_manager_ARN>" \
    --filter '{
```

```
    "principal": {
      "type": "GROUP",
      "id": "<GROUP_ID>"
    }
  }'
```

Sessões e controle de acesso

Noções básicas sobre sessões

Os usuários que acessam as Contas da AWS usando atribuições no gerenciador de acesso à conta iniciam os seguintes tipos de sessão:

1. Sessão de perfil do IAM — O perfil do IAM que um usuário assume é uma credencial temporária para acesso aos recursos da AWS com duração padrão de 1 hora.
2. Sessão do portal de acesso à conta — Os usuários iniciam essa sessão ao usar o portal de acesso à conta. Ela é gerenciada como uma sessão de aplicativo pelo IAM Identity Center.
3. Sessão do portal de acesso da AWS (também chamada de sessão interativa do usuário) — Essa sessão é gerenciada pelo IAM Identity Center.
4. Sessão de fonte de identidade externa (IdP ou Microsoft AD) — Essa sessão é aplicável quando a instância do IAM Identity Center está conectada a uma fonte de identidade externa para autenticação.

Para obter mais informações sobre sessões interativas com usuários e aplicativos no IAM Identity Center, incluindo a duração da sessão, consulte Noções básicas sobre sessões de autenticação no IAM Identity Center no Guia do usuário do Centro de Identidade do AWS IAM.

> **Important**
>
> As sessões de perfis do IAM atribuídas pelo gerenciador de acesso à conta operam com base na duração padrão da sessão de 1 hora, independentemente da duração máxima da sessão definida no perfil do IAM, do status das sessões no portal de acesso à conta e no portal de acesso da AWS.

Revogar acesso do usuário

Para revogar o acesso do usuário assumido por meio de atribuições de perfil do gerenciador de acesso à conta, consulte Revogar acesso do usuário no Guia do usuário do Centro de Identidade do AWS IAM.

Controle de acesso por atributo

Você pode configurar seu provedor de identidades (IdP) para enviar atributos como tags de sessão por meio de declarações SAML definindo o nome do atributo como `https://aws.amazon.com/SAML/Attributes/AccessControl:TagKey`, onde *`TagKey`* é a chave da tag de sessão que você deseja preencher. O IAM Identity Center passa o nome e o valor do atributo do IdP para avaliação da política.

Acesso da força de trabalho

O gerenciador de acesso à conta permite que você atribua acesso à Conta da AWS aos usuários e grupos da sua força de trabalho no IAM Identity Center, mapeando-os para os perfis do IAM nas suas Contas da AWS. Se você é iniciante no IAM Identity Center, consulte Configurar acesso da força de trabalho aos recursos da AWS no Guia do usuário do Centro de Identidade do AWS IAM para obter informações sobre sessões de autenticação, gerenciamento de acesso de usuários, gerenciamento de senhas e autenticação multifator.

Portais de usuários

Para acessar as contas da AWS atribuídas, os usuários da sua força de trabalho podem navegar diretamente até o portal de acesso à conta usando um marcador ou iniciá-lo dentro do portal de acesso da AWS.

Seus usuários podem interagir com dois portais de usuário:

Portal de acesso à conta

> Um portal (URL) que permite que os usuários da força de trabalho acessem as Contas da AWS atribuídas a eles usando o gerenciador de acesso à conta. O portal (URL) pode ser marcado como favorito para acesso direto e também está disponível no portal de acesso da AWS como um aplicativo gerenciado da AWS.

AWS Portal de acesso

Esse é o principal ponto de entrada na AWS para os usuários do IAM Identity Center. O portal exibe todos os aplicativos da AWS aos quais os usuários têm acesso, incluindo o portal de acesso à conta.

> **Tip**
>
> Recomendamos marcar os dois portais para uma navegação mais rápida. Se você usar um provedor de identidade externo, também poderá usá-lo para criar aplicativos de favoritos para os dois portais.

Conceitos básicos do portal de acesso à conta

Encontrar o URL do portal de acesso à conta

Para encontrar o URL do portal de acesso à conta, escolha a guia Configurações no console do gerenciador de acesso à conta. Consulte o URL do aplicativo na seção Detalhes da instância. O URL segue o padrão `https://[Tenant-ID].account-access.[Region].app.aws`. O ID do locatário é um identificador aleatório exclusivo, associado ao gerenciador de acesso à conta.

Se você usar a AWS CLI, poderá encontrar o ID do locatário executando o seguinte comando da CLI na conta gerencial ou de administrador delegado da sua organização.

```
aws account-access list-applications
```

A resposta contém informações sobre o gerenciador de acesso à conta, incluindo o ARN do aplicativo e o ID do locatário. Use o ID do locatário e o código da Região do gerenciador de acesso à conta para criar o URL do aplicativo. Siga o padrão fornecido anteriormente nesta seção.

```
{
    "applications": [
        {
            "applicationArn": "arn:aws:account-access:us-
west-2:123456789012:application/1234567890abcdef",
            "tenantId": "aa-gyxmap389",
            "createdAt": "2026-03-27T18:31:19+00:00",
```

```
            "updatedAt": "2026-03-27T18:31:19+00:00"
        }
    ]
}
```

Configurar um URL de portal personalizado

Para obter instruções sobre como implementar domínios personalizados, consulte Roteamento regional para portais de acesso da AWS: implementação de domínios personalizados para o IAM Identity Center no Blog de segurança da AWS. Você pode adaptar essas instruções ao padrão de URL do portal de acesso à conta.

Comunicar-se com seus usuários

Comunique o seguinte aos usuários da sua força de trabalho:

- O URL do portal de acesso à conta.
- O portal de acesso à conta também está disponível para seus usuários como um aplicativo na guia Aplicativos no portal de acesso da AWS.
- Se você habilitou o lançamento do portal de acesso à conta com base em um portal externo do provedor de identidade ou de outro portal fora da AWS, inclua essas informações na comunicação.

Acesso dos usuários a Contas da AWS

Este tópico descreve como os usuários acessam as Contas da AWS atribuídas a eles com o gerenciador de acesso à conta.

Obter o URL do portal

Existem várias maneiras para acessar as Contas da AWS atribuídas a você:

1. Seu administrador pode lhe fornecer um URL do portal de acesso à conta (por exemplo, `https://aa-gyxmap389.account-access.us-east-2.app.aws` ou um URL personalizado).
2. Seu administrador pode fornecer o URL do portal de acesso principal da AWS, que exibe todos os aplicativos da AWS atribuídos a você, incluindo o portal de acesso à conta.
3. Sua organização pode ter permitido que você iniciasse o portal de acesso à conta com base em um portal externo do provedor de identidade ou de outro portal fora da AWS.

Acessar as Contas da AWS com um navegador da web

O portal de acesso à conta exibe as Contas da AWS às quais você tem acesso e os perfis do IAM que você pode assumir em cada conta.

Para acessar uma conta em um navegador

1. Navegue até o URL do portal de acesso à conta e, quando solicitado, faça login usando suas credenciais conforme as instruções do administrador.

   > **Note**
   >
   > Para iniciar o portal de acesso à conta com base no portal de acesso da AWS, navegue até a guia Aplicativos.

2. A página do navegador exibe as Contas da AWS atribuídas a você. Você pode pesquisar as contas por nome ou ID da conta.
3. Expanda a conta de interesse para ver todos os perfis que você pode assumir na conta.
4. Escolha o perfil do IAM com o qual fazer login (por exemplo, Desenvolvedor). Uma nova guia do navegador é aberta, na qual você fará o login com o perfil escolhido.

Acessar as Contas da AWS com a AWS CLI ou SDKs da AWS

Você pode acessar os serviços AWS programaticamente usando o AWS Command Line Interface ou os kits de desenvolvimento de software (SDKs) AWS com credenciais de usuário do IAM Identity Center.

O portal de acesso à conta fornece aos usuários acesso de single sign-on às Contas da AWS para atribuições de conta administradas no gerenciador de acesso à conta. Depois de entrar no portal de acesso à conta, você pode obter credenciais temporárias. Em seguida, pode usar as credenciais na AWS CLI ou SDKs da AWS para acessar recursos em uma Conta da AWS.

Se você estiver usando a AWS CLI para acessar os serviços da AWS de forma programática, você pode usar os procedimentos neste tópico para iniciar o acesso ao AWS CLI. Para obter informações sobre o AWS CLI, consulte o Guia do usuário do AWS Command Line Interface.

Se você estiver usando os SDKs da AWS para acessar os serviços da AWS de forma programática, siga os procedimentos neste tópico para também estabelecer diretamente a autenticação dos SDKs

da AWS. Para obter informações sobre os SDKs AWS, consulte o Guia de referência de SDKs AWS e ferramentas.

Acesso à Conta da AWS com o aws login (recomendado)

Você pode usar o comando aws login para obter acesso às Contas da AWS atribuídas a você no gerenciador de acesso à conta. Primeiro, instale a AWS CLI mais recente. Depois, siga este procedimento:

Para acessar uma conta com a AWS CLI

1. No seu navegador padrão, faça login no portal de acesso à conta e, em seguida, acesse a Conta da AWS de destino usando um dos perfis do IAM atribuídos à conta. Siga o procedimento descrito anteriormente neste tópico.
2. Na linha de comando, execute o comando aws login.
3. Se você não tiver definido uma Região padrão, a AWS CLI solicitará que você especifique a Região da AWS de sua escolha (por exemplo, us-east-2). A AWS CLI memorizará sua escolha para uso futuro.
4. A AWS CLI abre uma página da web em seu navegador padrão e solicita que você continue com a sessão ativa que acabou de iniciar ou entre em uma nova sessão.
5. Na página da web, escolha o bloco com a sessão que você iniciou anteriormente no navegador.
6. A página do navegador confirma que suas credenciais foram compartilhadas com sucesso e podem ser usadas até que sua sessão expire. Agora, você pode fechar a guia com a página da web e continuar trabalhando com a AWS CLI usando esta sessão.

> **Note**
>
> Comandos daAWS CLI específicos do IAM Identity Center, como aws sso configure e aws sso login, não funcionam com atribuições de conta no gerenciador de acesso à conta.

Acesso à Conta da AWS com chaves de acesso

Você pode usar o método de atualização manual de credenciais para obter credenciais temporárias para um perfil do IAM atribuído a você em uma Conta da AWS específica. Para fazer isso, você copia e cola os comandos necessários para as credenciais temporárias. Com esse método, você deve atualizar as credenciais temporárias manualmente.

Você pode executar comandos AWS CLI até que suas credenciais temporárias expirem.

Para obter credenciais que você atualiza manualmente

1. Entre no portal de acesso à conta conforme explicado em Acessar as Contas da AWS com um navegador da web.
2. Localize a Conta da AWS da qual você deseja recuperar as credenciais de acesso e a expanda para mostrar o nome do perfil do IAM (por exemplo, Desenvolvedor).
3. Escolha Gerar chave de acesso ao lado do perfil desejado.

   > **Note**
   >
   > Se você não vir nenhuma atribuição de Conta da AWS, é provável que nenhuma conta tenha sido atribuída a você. Nesse caso, entre em contato com seu administrador e peça para atribuir o acesso para você.

4. Na caixa de diálogo Obter credenciais, escolha a guia MacOS e Linux, Windows ou PowerShell, dependendo do sistema operacional no qual você instalou a AWS CLI.
5. Escolha uma das seguintes opções:

   - Opção 1: definir variáveis de ambiente AWS

     Escolha essa opção para substituir todas as configurações de credenciais, inclusive aquelas nos arquivos de credenciais e nos de configuração. Para obter mais informações, consulte Variáveis de ambiente para configurar no AWS CLI no Guia do usuário AWS Command Line Interface.

     Para usar essa opção, copie os comandos na área de transferência, cole os comandos na janela do terminal AWS CLI e pressione Enter para definir as variáveis de ambiente necessárias.

   - Opção 2: use valores individuais em seu cliente de serviço AWS

     Escolha essa opção para acessar os recursos AWS de um cliente de serviço AWS. Para obter mais informações, consulte Ferramentas para criar na AWS.

     Para usar essa opção, copie os valores para sua área de transferência, cole os valores em seu código e atribua-os às variáveis apropriadas para seu SDK. Para obter mais informações, consulte a documentação específica para sua API do SDK.

Criar links de atalho para destinos no Console de Gerenciamento da AWS

O portal de acesso à conta oferece a opção de criar um link direto compartilhável.

Para criar um link de atalho

1. Depois de fazer login no portal de acesso à conta, escolha o botão Criar link de atalho.
2. Na caixa de diálogo:

    a. Escolha uma Conta da AWS usando o ID da conta ou o nome da conta. À medida que você digita, um menu suspenso exibe IDs e nomes de contas correspondentes que você pode acessar. Você só pode escolher uma conta à qual tenha acesso.

    b. (Opcional) Escolha um perfil do IAM na lista suspensa. Esses são os perfis do IAM atribuídos a você para a conta selecionada. Se você não escolher o perfil, será solicitado que os usuários selecionem um perfil atribuído a eles para a conta escolhida ao usar o link de atalho.

> **Note**
>
> Você não pode conceder novos acessos com links de atalho. Os links de atalho só funcionam com as contas e os perfis já atribuídos ao usuário.

Criar links de atalho seguros do Console de Gerenciamento da AWS com codificação de URL

Todos os valores de parâmetros do URL, incluindo o ID da conta, o nome do perfil do IAM e o URL de destino, devem ser codificados no URL.

Os links de atalho estendem o URL do portal de acesso à conta com o seguinte caminho:

```
/#/console?account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

O URL completo em uma Região comercial da AWS segue este padrão:

```
https://[Tenant-ID].account-access.[Region].app.aws/#/console?
account_id=[account_ID]&role_name=[role_name]&destination=[destination_URL]
```

Aqui está um exemplo de link de atalho que conecta um usuário à conta 123456789012 com o nome do perfil do IAM S3FullAccess e o leva à página inicial do console S3:

```
https://aa-gyxmap389.account-access.us-east-2.app.aws/#/console?
account_id=123456789012&role_name=S3FullAccess&destination=https%3A%2F
%2Fconsole.aws.amazon.com%2Fs3%2Fhome
```

Considerações

As considerações a seguir se aplicam ao uso de links de atalho:

- Se você compartilhar links de atalho com colegas de trabalho, eles precisam ter as mesmas atribuições de conta para poder usar os links da mesma forma.
- Os links de atalho são Regionais. Se a região referenciada estiver passando por uma interrupção, o link poderá não funcionar.
- Os links de atalho contêm o ID da conta e o nome do perfil em texto simples. Considere se as políticas da sua organização restringem onde esses metadados são armazenados.

Segurança no gerenciador de acesso à conta

Esta seção fornece informações de segurança específicas para o gerenciador de acesso à conta. Ele complementa as informações contidas em Segurança, que abrangem tópicos de segurança aplicáveis ao IAM como um todo.

Gerenciamento de identidade e acesso para o gerenciador de acesso à conta

As seções abaixo descrevem as permissões necessárias para você:

- Administrar o gerenciador de acesso à conta na conta gerencial da sua organização
- Administrar o gerenciador de acesso à conta com base em uma conta de administrador delegado

Administrar o gerenciador de acesso à conta com base na conta gerencial da sua organização

Para administrar as configurações do gerenciador de acesso à conta na conta gerencial da organização na AWS, você precisa das seguintes permissões:

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "AAMCreateApplication",
      "Effect": "Allow",
      "Action": "account-access:CreateApplication",
      "Resource": "*"
    },
    {
      "Sid": "AAMApplicationAndEntitlementManagement",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteApplication",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:CreateApplication",
        "sso:DeleteApplication",
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances",
        "sso:PutApplicationAccessScope",
        "sso:PutApplicationAssignmentConfiguration",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant"
      ],
```

```
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentKMSActions",
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "*"
    },
    {
      "Sid": "CreateServiceLinkedRole",
      "Effect": "Allow",
```

```
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "arn:aws:iam::111122223333:role/aws-service-role/account-
access.amazonaws.com/AWSServiceRoleForAccountAccessManager",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "account-access.amazonaws.com"
        }
      }
    }
  ]
}
```

Opcional: para restringir `account-access:CreateApplication` por tag, substitua a instrução AAMCreateApplication por:

```
{
  "Sid": "AAMCreateApplicationConstrainedByTag",
  "Effect": "Allow",
  "Action": "account-access:CreateApplication",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:RequestTag/Environment": "Production"
    },
    "ForAllValues:StringEquals": {
      "aws:TagKeys": [
        "Environment"
      ]
    }
  }
}
```

Opcional: para gerenciar e registrar administradores delegados para o gerenciador de acesso à conta, adicione:

```
{
  "Sid": "ManageDelegatedAdministrator",
  "Effect": "Allow",
  "Action": [
    "organizations:DeregisterDelegatedAdministrator",
    "organizations:RegisterDelegatedAdministrator"
  ],
  "Resource": "*",
```

```
  "Condition": {
    "StringEquals": {
      "organizations:ServicePrincipal": "account-access.amazonaws.com"
    }
  }
}
```

Para atribuir perfis do IAM aos usuários e grupos da sua força de trabalho com base na conta gerencial da organização da AWS sem administrar as configurações do gerenciador de acesso à conta, você precisa das seguintes permissões:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
```

```
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Administrar o gerenciador de acesso à conta em uma conta de administração delegada

Para administrar as configurações do gerenciador de acesso à conta com base em uma conta de administração delegada, você precisa das seguintes permissões:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMDelegatedAdminManagement",
      "Effect": "Allow",
```

```
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements",
        "account-access:ListTagsForResource",
        "account-access:TagResource",
        "account-access:UntagResource"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": [
        "sso:DescribeApplication",
        "sso:DescribeInstance",
        "sso:DescribeRegion",
        "sso:ListInstances"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
```

```
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Para atribuir perfis do IAM aos usuários e grupos da força de trabalho com base em uma conta de administração delegada sem administrar as configurações do gerenciador de acesso à conta, você precisa das seguintes permissões:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AAMManageEntitlementsOnly",
      "Effect": "Allow",
      "Action": [
        "account-access:CreateEntitlement",
        "account-access:DeleteEntitlement",
        "account-access:GetApplication",
        "account-access:GetEntitlement",
        "account-access:ListEntitlements"
      ],
      "Resource": "arn:aws:account-access:*:111122223333:application/*"
    },
    {
      "Sid": "AAMListApplications",
      "Effect": "Allow",
```

```
      "Action": "account-access:ListApplications",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityCenterActions",
      "Effect": "Allow",
      "Action": "sso:ListInstances",
      "Resource": "*"
    },
    {
      "Sid": "DependentIdentityStoreActions",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeGroup",
        "identitystore:DescribeUser",
        "identitystore:ListGroupMemberships",
        "sso-directory:DescribeGroup",
        "sso-directory:DescribeGroups",
        "sso-directory:DescribeUser",
        "sso-directory:DescribeUsers",
        "sso-directory:ListGroupsForUser",
        "sso-directory:ListMembersInGroup",
        "sso-directory:SearchGroups",
        "sso-directory:SearchUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DependentOrganizationsActions",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

Proteção de dados no gerenciador de acesso à conta

Esta seção complementa as informações fornecidas em Proteção de dados no AWS Identity and Access Management com detalhes adicionais específicos para o gerenciador de acesso à conta.

Uso de dados de usuário e grupo do IAM Identity Center

O gerenciador de acesso à conta usa os dados de usuários e grupos do IAM Identity Center para gerenciar as atribuições da conta. No entanto, o gerenciador de acesso à conta armazena e gera os logs somente de identificadores gerados pela AWS (ID de usuário e ID de grupo), e não nomes de usuário, endereços de e-mail ou outros atributos pessoais.

Geração de logs e monitoramento no gerenciador de acesso à conta

O gerenciador de acesso à conta usa seu próprio namespace de API (`account-access`) e gera os logs das chamadas de API por meio do CloudTrail.

Para obter mais informações sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.

Cotas e limites no gerenciador de acesso à conta

As tabelas a seguir descrevem as cotas dentro do gerenciador de acesso à conta. As solicitações de aumento de cota devem vir de uma conta administrativa ou de administrador delegado. Para aumentar uma cota, consulte solicitação de aumento de cota.

Cota de atribuição de grupo

Recurso	Cota padrão	Pode ser aumentada
Número máximo de grupos atribuídos a um perfil do IAM	20	Sim

Limites de controle de utilização

Recurso	Cota padrão	Pode ser aumentada
API de acesso à conta	A API de acesso à conta tem um limite de controle de utilização coletivo de 20 transações por segundo (TPS). Para APIs de leitura, você pode abrir um caso de suporte para solicitar um aumento de limite. As operações da API de gravação têm um limite de 15 chamadas assíncronas pendentes. Este limite não pode ser aumentado.	Sim

Exemplos de código para o IAM usando AWS SDKs

Os exemplos de código a seguir mostram como usar o IAM com um Kit de Desenvolvimento de Software (SDK) da AWS.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Exemplos de código para o IAM usando AWS SDKs
  - Exemplos básicos do IAM usando SDKs da AWS
    - Olá, IAM
    - Conhecer os conceitos básicos do IAM com um AWS SDK
    - Ações do IAM usando AWS SDKs
      - Usar AddClientIdToOpenIdConnectProvider com uma CLI
      - Usar AddRoleToInstanceProfile com uma CLI
      - Usar AddUserToGroup com uma CLI
      - Usar AttachGroupPolicy com uma CLI
      - Usar AttachRolePolicy com o AWS SDK ou a CLI
      - Usar AttachUserPolicy com o AWS SDK ou a CLI
      - Usar ChangePassword com uma CLI
      - Usar CreateAccessKey com o AWS SDK ou a CLI
      - Usar CreateAccountAlias com o AWS SDK ou a CLI
      - Usar CreateGroup com o AWS SDK ou a CLI
      - Usar CreateInstanceProfile com o AWS SDK ou a CLI
      - Usar CreateLoginProfile com uma CLI
      - Usar CreateOpenIdConnectProvider com uma CLI
      - Usar CreatePolicy com o AWS SDK ou a CLI
      - Usar CreatePolicyVersion com o AWS SDK ou a CLI
      - Usar CreateRole com o AWS SDK ou a CLI
      - Usar CreateSAMLProvider com o AWS SDK ou a CLI

- Usar CreateServiceLinkedRole com o AWS SDK ou a CLI
- Usar CreateUser com o AWS SDK ou a CLI
- Usar CreateVirtualMfaDevice com uma CLI
- Usar DeactivateMfaDevice com uma CLI
- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com uma CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com uma CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com uma CLI
- Usar DeleteOpenIdConnectProvider com uma CLI
- Usar DeletePolicy com o AWS SDK ou a CLI
- Usar DeletePolicyVersion com o AWS SDK ou a CLI
- Usar DeleteRole com o AWS SDK ou a CLI
- Usar DeleteRolePermissionsBoundary com uma CLI
- Usar DeleteRolePolicy com o AWS SDK ou a CLI
- Usar DeleteSAMLProvider com o AWS SDK ou a CLI
- Usar DeleteServerCertificate com o AWS SDK ou a CLI
- Usar DeleteServiceLinkedRole com o AWS SDK ou a CLI
- Usar DeleteSigningCertificate com uma CLI
- Usar DeleteUser com o AWS SDK ou a CLI
- Usar DeleteUserPermissionsBoundary com uma CLI
- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com uma CLI
- Usar DetachGroupPolicy com uma CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI
- Usar EnableMfaDevice com uma CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI

- Usar GenerateServiceLastAccessedDetails com uma CLI
- Usar GetAccessKeyLastUsed com o AWS SDK ou a CLI
- Usar GetAccountAuthorizationDetails com o AWS SDK ou a CLI
- Usar GetAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar GetAccountSummary com o AWS SDK ou a CLI
- Usar GetContextKeysForCustomPolicy com uma CLI
- Usar GetContextKeysForPrincipalPolicy com uma CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com uma CLI
- Usar GetGroupPolicy com uma CLI
- Usar GetInstanceProfile com uma CLI
- Usar GetLoginProfile com uma CLI
- Usar GetOpenIdConnectProvider com uma CLI
- Usar GetPolicy com o AWS SDK ou a CLI
- Usar GetPolicyVersion com o AWS SDK ou a CLI
- Usar GetRole com o AWS SDK ou a CLI
- Usar GetRolePolicy com uma CLI
- Usar GetSamlProvider com uma CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com uma CLI
- Usar GetServiceLastAccessedDetailsWithEntities com uma CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI
- Usar GetUser com o AWS SDK ou a CLI
- Usar GetUserPolicy com uma CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com uma CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI
- Usar ListAttachedUserPolicies com uma CLI
- Usar ListEntitiesForPolicy com uma CLI

- Usar ListGroupPolicies com uma CLI
- Usar ListGroups com o AWS SDK ou a CLI
- Usar ListGroupsForUser com uma CLI
- Usar ListInstanceProfiles com uma CLI
- Usar ListInstanceProfilesForRole com uma CLI
- Usar ListMfaDevices com uma CLI
- Usar ListOpenIdConnectProviders com uma CLI
- Usar ListPolicies com o AWS SDK ou a CLI
- Usar ListPolicyVersions com o AWS SDK ou a CLI
- Usar ListRolePolicies com o AWS SDK ou a CLI
- Usar ListRoleTags com uma CLI
- Usar ListRoles com o AWS SDK ou a CLI
- Usar ListSAMLProviders com o AWS SDK ou a CLI
- Usar ListServerCertificates com o AWS SDK ou a CLI
- Usar ListSigningCertificates com uma CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com uma CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com uma CLI
- Usar PutGroupPolicy com uma CLI
- Usar PutRolePermissionsBoundary com uma CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com uma CLI
- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com uma CLI
- Usar RemoveRoleFromInstanceProfile com uma CLI
- Usar RemoveUserFromGroup com uma CLI
- Usar ResyncMfaDevice com uma CLI
- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com uma CLI

- Usar TagUser com uma CLI
- Usar UntagRole com uma CLI
- Usar UntagUser com uma CLI
- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com uma CLI
- Usar UpdateAssumeRolePolicy com uma CLI
- Usar UpdateGroup com uma CLI
- Usar UpdateLoginProfile com uma CLI
- Usar UpdateOpenIdConnectProviderThumbprint com uma CLI
- Usar UpdateRole com uma CLI
- Usar UpdateRoleDescription com uma CLI
- Usar UpdateSamlProvider com uma CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com uma CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com uma CLI

- Cenários para o IAM usando AWS SDKs
  - Criar e gerenciar um serviço resiliente usando um AWS SDK
  - Configurar a conectividade do serviço de contêiner
  - Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
  - Criar uma API REST com integração de proxy de função
  - Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK
  - Como criar um serviço de contêiner para instâncias de máquinas virtuais
  - Como criar um espaço de trabalho de monitoramento gerenciado
  - Como criar um painel de monitoramento com o nome da função como uma variável
  - Como criar sua primeira função com tecnologia sem servidor
  - Conceitos básicos de data warehouses com tecnologia sem servidor usando a CLI
  - Conceitos básicos do gerenciamento de configurações
  - Conceitos básicos da proteção de dispositivos da Internet das Coisas

- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos de clusters de data warehouse provisionados
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar chaves de acesso do IAM usando um AWS SDK
- Gerenciar políticas do IAM usando um AWS SDK
- Gerenciar perfis do IAM usando um AWS SDK
- Gerenciar a conta do IAM usando um AWS SDK
- Como mover segredos codificados para o armazenamento seguro de segredos
- Permite que o atributo de automação AWS Compute Optimizer aplique as ações recomendadas
- Política para permitir a automação em toda a sua organização
- Política para permitir a automação na sua conta
- Política para conceder acesso total ao Compute Optimizer Automation para uma conta gerencial de uma organização
- Política para conceder acesso total ao Compute Optimizer Automation para contas autônomas da AWS
- Política para conceder acesso somente leitura ao Compute Optimizer Automation para uma conta gerencial de uma organização
- Política para conceder acesso somente leitura ao Compute Optimizer Automation para contas autônomas da AWS
- Política para conceder permissões de perfil vinculado ao serviço para o Compute Optimization Automation
- Reverter uma versão de política do IAM usando um AWS SDK
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Configurar o controle de acesso por atributo para o DynamoDB usando a AWS Command Line Interface v2
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor

        - Trabalhar com fluxos e vida útil do DynamoDB usando a AWS Command Line Interface v2
        - Trabalhar com a API IAM Policy Builder usando um AWS SDK
- Exemplos de código para o AWS STS usando AWS SDKs
    - Exemplos básicos para o AWS STS usando SDKs da AWS
        - Ações para o AWS STS usando AWS SDKs
            - Usar AssumeRole com o AWS SDK ou a CLI
            - Usar AssumeRoleWithWebIdentity com uma CLI
            - Usar DecodeAuthorizationMessage com uma CLI
            - Usar GetFederationToken com uma CLI
            - Usar GetSessionToken com o AWS SDK ou a CLI
    - Cenários para o AWS STS usando AWS SDKs
        - Assumir um perfil do IAM que exija um token de MFA com o AWS STS usando um AWS SDK
        - Configurar a conectividade do serviço de contêiner
        - Crie uma URL com o AWS STS para usuários federados usando um AWS SDK
        - Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
        - Criar uma API REST com integração de proxy de função
        - Como criar um serviço de contêiner para instâncias de máquinas virtuais
        - Como criar um espaço de trabalho de monitoramento gerenciado
        - Como criar um painel de monitoramento com o nome da função como uma variável
        - Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK
        - Conceitos básicos do gerenciamento de configurações
        - Conceitos básicos de registros de contêineres
        - Conceitos básicos de mensagens da Internet das Coisas
        - Conceitos básicos de arquivos de atributos de machine learning
        - Conceitos básicos de clusters gerenciados de kubernetes
        - Conceitos básicos de streaming gerenciado
        - Conceitos básicos das notificações push
        - Conceitos básicos dos mecanismos de pesquisa e analytics
        - Conceitos básicos de firewalls do aplicativo Web

- Como mover segredos codificados para o armazenamento seguro de segredos
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas

Exemplos de código para o IAM usando AWS SDKs

Os exemplos de código a seguir mostram como usar o IAM com um Kit de Desenvolvimento de Software (SDK) da AWS.

As noções básicas são exemplos de código que mostram como realizar as operações essenciais em um serviço.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para ver uma lista completa dos Guias do desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

IAM

- Exemplos básicos do IAM usando SDKs da AWS
  - Olá, IAM
  - Conhecer os conceitos básicos do IAM com um AWS SDK
  - Ações do IAM usando AWS SDKs
    - Usar AddClientIdToOpenIdConnectProvider com uma CLI
    - Usar AddRoleToInstanceProfile com uma CLI
    - Usar AddUserToGroup com uma CLI
    - Usar AttachGroupPolicy com uma CLI
    - Usar AttachRolePolicy com o AWS SDK ou a CLI
    - Usar AttachUserPolicy com o AWS SDK ou a CLI
    - Usar ChangePassword com uma CLI

- Usar CreateAccessKey com o AWS SDK ou a CLI
- Usar CreateAccountAlias com o AWS SDK ou a CLI
- Usar CreateGroup com o AWS SDK ou a CLI
- Usar CreateInstanceProfile com o AWS SDK ou a CLI
- Usar CreateLoginProfile com uma CLI
- Usar CreateOpenIdConnectProvider com uma CLI
- Usar CreatePolicy com o AWS SDK ou a CLI
- Usar CreatePolicyVersion com o AWS SDK ou a CLI
- Usar CreateRole com o AWS SDK ou a CLI
- Usar CreateSAMLProvider com o AWS SDK ou a CLI
- Usar CreateServiceLinkedRole com o AWS SDK ou a CLI
- Usar CreateUser com o AWS SDK ou a CLI
- Usar CreateVirtualMfaDevice com uma CLI
- Usar DeactivateMfaDevice com uma CLI
- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com uma CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com uma CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com uma CLI
- Usar DeleteOpenIdConnectProvider com uma CLI
- Usar DeletePolicy com o AWS SDK ou a CLI
- Usar DeletePolicyVersion com o AWS SDK ou a CLI
- Usar DeleteRole com o AWS SDK ou a CLI
- Usar DeleteRolePermissionsBoundary com uma CLI
- Usar DeleteRolePolicy com o AWS SDK ou a CLI
- Usar DeleteSAMLProvider com o AWS SDK ou a CLI
- Usar DeleteServerCertificate com o AWS SDK ou a CLI
- Usar DeleteServiceLinkedRole com o AWS SDK ou a CLI

- Usar DeleteSigningCertificate com uma CLI
- Usar DeleteUser com o AWS SDK ou a CLI
- Usar DeleteUserPermissionsBoundary com uma CLI
- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com uma CLI
- Usar DetachGroupPolicy com uma CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI
- Usar EnableMfaDevice com uma CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI
- Usar GenerateServiceLastAccessedDetails com uma CLI
- Usar GetAccessKeyLastUsed com o AWS SDK ou a CLI
- Usar GetAccountAuthorizationDetails com o AWS SDK ou a CLI
- Usar GetAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar GetAccountSummary com o AWS SDK ou a CLI
- Usar GetContextKeysForCustomPolicy com uma CLI
- Usar GetContextKeysForPrincipalPolicy com uma CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com uma CLI
- Usar GetGroupPolicy com uma CLI
- Usar GetInstanceProfile com uma CLI
- Usar GetLoginProfile com uma CLI
- Usar GetOpenIdConnectProvider com uma CLI
- Usar GetPolicy com o AWS SDK ou a CLI
- Usar GetPolicyVersion com o AWS SDK ou a CLI
- Usar GetRole com o AWS SDK ou a CLI
- Usar GetRolePolicy com uma CLI
- Usar GetSamlProvider com uma CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com uma CLI

- Usar GetServiceLastAccessedDetailsWithEntities com uma CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI
- Usar GetUser com o AWS SDK ou a CLI
- Usar GetUserPolicy com uma CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com uma CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI
- Usar ListAttachedUserPolicies com uma CLI
- Usar ListEntitiesForPolicy com uma CLI
- Usar ListGroupPolicies com uma CLI
- Usar ListGroups com o AWS SDK ou a CLI
- Usar ListGroupsForUser com uma CLI
- Usar ListInstanceProfiles com uma CLI
- Usar ListInstanceProfilesForRole com uma CLI
- Usar ListMfaDevices com uma CLI
- Usar ListOpenIdConnectProviders com uma CLI
- Usar ListPolicies com o AWS SDK ou a CLI
- Usar ListPolicyVersions com o AWS SDK ou a CLI
- Usar ListRolePolicies com o AWS SDK ou a CLI
- Usar ListRoleTags com uma CLI
- Usar ListRoles com o AWS SDK ou a CLI
- Usar ListSAMLProviders com o AWS SDK ou a CLI
- Usar ListServerCertificates com o AWS SDK ou a CLI
- Usar ListSigningCertificates com uma CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com uma CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com uma CLI
- Usar PutGroupPolicy com uma CLI

- Usar PutRolePermissionsBoundary com uma CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com uma CLI
- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com uma CLI
- Usar RemoveRoleFromInstanceProfile com uma CLI
- Usar RemoveUserFromGroup com uma CLI
- Usar ResyncMfaDevice com uma CLI
- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com uma CLI
- Usar TagUser com uma CLI
- Usar UntagRole com uma CLI
- Usar UntagUser com uma CLI
- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com uma CLI
- Usar UpdateAssumeRolePolicy com uma CLI
- Usar UpdateGroup com uma CLI
- Usar UpdateLoginProfile com uma CLI
- Usar UpdateOpenIdConnectProviderThumbprint com uma CLI
- Usar UpdateRole com uma CLI
- Usar UpdateRoleDescription com uma CLI
- Usar UpdateSamlProvider com uma CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com uma CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com uma CLI

- Cenários para o IAM usando AWS SDKs
  - Criar e gerenciar um serviço resiliente usando um AWS SDK
  - Configurar a conectividade do serviço de contêiner

- Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
- Criar uma API REST com integração de proxy de função
- Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Como criar sua primeira função com tecnologia sem servidor
- Conceitos básicos de data warehouses com tecnologia sem servidor usando a CLI
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos de clusters de data warehouse provisionados
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar chaves de acesso do IAM usando um AWS SDK
- Gerenciar políticas do IAM usando um AWS SDK
- Gerenciar perfis do IAM usando um AWS SDK
- Gerenciar a conta do IAM usando um AWS SDK
- Como mover segredos codificados para o armazenamento seguro de segredos
- Permite que o atributo de automação AWS Compute Optimizer aplique as ações recomendadas
- Política para permitir a automação em toda a sua organização
- Política para permitir a automação na sua conta
- Política para conceder acesso total ao Compute Optimizer Automation para uma conta gerencial de uma organização
- Política para conceder acesso total ao Compute Optimizer Automation para contas autônomas da AWS
- Política para conceder acesso somente leitura ao Compute Optimizer Automation para uma conta gerencial de uma organização
- Política para conceder acesso somente leitura ao Compute Optimizer Automation para contas autônomas da AWS

- Política para conceder permissões de perfil vinculado ao serviço para o Compute Optimization Automation
- Reverter uma versão de política do IAM usando um AWS SDK
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Configurar o controle de acesso por atributo para o DynamoDB usando a AWS Command Line Interface v2
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor
- Trabalhar com fluxos e vida útil do DynamoDB usando a AWS Command Line Interface v2
- Trabalhar com a API IAM Policy Builder usando um AWS SDK

Exemplos básicos do IAM usando SDKs da AWS

Os exemplos de código a seguir mostram como usar os conceitos básicos do AWS Identity and Access Management com SDKs da AWS.

Exemplos

- Olá, IAM
- Conhecer os conceitos básicos do IAM com um AWS SDK
- Ações do IAM usando AWS SDKs
  - Usar AddClientIdToOpenIdConnectProvider com uma CLI
  - Usar AddRoleToInstanceProfile com uma CLI
  - Usar AddUserToGroup com uma CLI
  - Usar AttachGroupPolicy com uma CLI
  - Usar AttachRolePolicy com o AWS SDK ou a CLI
  - Usar AttachUserPolicy com o AWS SDK ou a CLI
  - Usar ChangePassword com uma CLI
  - Usar CreateAccessKey com o AWS SDK ou a CLI
  - Usar CreateAccountAlias com o AWS SDK ou a CLI
  - Usar CreateGroup com o AWS SDK ou a CLI

- Usar CreateInstanceProfile com o AWS SDK ou a CLI
- Usar CreateLoginProfile com uma CLI
- Usar CreateOpenIdConnectProvider com uma CLI
- Usar CreatePolicy com o AWS SDK ou a CLI
- Usar CreatePolicyVersion com o AWS SDK ou a CLI
- Usar CreateRole com o AWS SDK ou a CLI
- Usar CreateSAMLProvider com o AWS SDK ou a CLI
- Usar CreateServiceLinkedRole com o AWS SDK ou a CLI
- Usar CreateUser com o AWS SDK ou a CLI
- Usar CreateVirtualMfaDevice com uma CLI
- Usar DeactivateMfaDevice com uma CLI
- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com uma CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com uma CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com uma CLI
- Usar DeleteOpenIdConnectProvider com uma CLI
- Usar DeletePolicy com o AWS SDK ou a CLI
- Usar DeletePolicyVersion com o AWS SDK ou a CLI
- Usar DeleteRole com o AWS SDK ou a CLI
- Usar DeleteRolePermissionsBoundary com uma CLI
- Usar DeleteRolePolicy com o AWS SDK ou a CLI
- Usar DeleteSAMLProvider com o AWS SDK ou a CLI
- Usar DeleteServerCertificate com o AWS SDK ou a CLI
- Usar DeleteServiceLinkedRole com o AWS SDK ou a CLI
- Usar DeleteSigningCertificate com uma CLI
- Usar DeleteUser com o AWS SDK ou a CLI

- Usar DeleteUserPermissionsBoundary com uma CLI

- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com uma CLI
- Usar DetachGroupPolicy com uma CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI
- Usar EnableMfaDevice com uma CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI
- Usar GenerateServiceLastAccessedDetails com uma CLI
- Usar GetAccessKeyLastUsed com o AWS SDK ou a CLI
- Usar GetAccountAuthorizationDetails com o AWS SDK ou a CLI
- Usar GetAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar GetAccountSummary com o AWS SDK ou a CLI
- Usar GetContextKeysForCustomPolicy com uma CLI
- Usar GetContextKeysForPrincipalPolicy com uma CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com uma CLI
- Usar GetGroupPolicy com uma CLI
- Usar GetInstanceProfile com uma CLI
- Usar GetLoginProfile com uma CLI
- Usar GetOpenIdConnectProvider com uma CLI
- Usar GetPolicy com o AWS SDK ou a CLI
- Usar GetPolicyVersion com o AWS SDK ou a CLI
- Usar GetRole com o AWS SDK ou a CLI
- Usar GetRolePolicy com uma CLI
- Usar GetSamlProvider com uma CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com uma CLI
- Usar GetServiceLastAccessedDetailsWithEntities com uma CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI

- Usar GetUser com o AWS SDK ou a CLI

- Usar GetUserPolicy com uma CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com uma CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI
- Usar ListAttachedUserPolicies com uma CLI
- Usar ListEntitiesForPolicy com uma CLI
- Usar ListGroupPolicies com uma CLI
- Usar ListGroups com o AWS SDK ou a CLI
- Usar ListGroupsForUser com uma CLI
- Usar ListInstanceProfiles com uma CLI
- Usar ListInstanceProfilesForRole com uma CLI
- Usar ListMfaDevices com uma CLI
- Usar ListOpenIdConnectProviders com uma CLI
- Usar ListPolicies com o AWS SDK ou a CLI
- Usar ListPolicyVersions com o AWS SDK ou a CLI
- Usar ListRolePolicies com o AWS SDK ou a CLI
- Usar ListRoleTags com uma CLI
- Usar ListRoles com o AWS SDK ou a CLI
- Usar ListSAMLProviders com o AWS SDK ou a CLI
- Usar ListServerCertificates com o AWS SDK ou a CLI
- Usar ListSigningCertificates com uma CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com uma CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com uma CLI
- Usar PutGroupPolicy com uma CLI
- Usar PutRolePermissionsBoundary com uma CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com uma CLI

- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com uma CLI
- Usar RemoveRoleFromInstanceProfile com uma CLI
- Usar RemoveUserFromGroup com uma CLI
- Usar ResyncMfaDevice com uma CLI
- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com uma CLI
- Usar TagUser com uma CLI
- Usar UntagRole com uma CLI
- Usar UntagUser com uma CLI
- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com uma CLI
- Usar UpdateAssumeRolePolicy com uma CLI
- Usar UpdateGroup com uma CLI
- Usar UpdateLoginProfile com uma CLI
- Usar UpdateOpenIdConnectProviderThumbprint com uma CLI
- Usar UpdateRole com uma CLI
- Usar UpdateRoleDescription com uma CLI
- Usar UpdateSamlProvider com uma CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com uma CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com uma CLI

Olá, IAM

O exemplo de código a seguir mostra como começar a usar o IAM.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para .NET.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Código para o arquivo CMakeLists.txt do CMake.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Código para o arquivo de origem iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para C++.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
```

```
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
```

```
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      for (const policy of page.Policies) {
        console.log(`${policy.PolicyName}`);
        policyCount++;
      }
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para JavaScript.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import boto3


def main():
    """
    Lists the managed policies in your AWS account using the AWS SDK for Python
 (Boto3).
    """
    iam = boto3.client("iam")

    try:
        # Get a paginator for the list_policies operation
        paginator = iam.get_paginator("list_policies")

        # Iterate through the pages of results
        for page in paginator.paginate(Scope="All", OnlyAttached=False):
            for policy in page["Policies"]:
                print(f"Policy name: {policy['PolicyName']}")
                print(f"  Policy ARN: {policy['Arn']}")
    except boto3.exceptions.BotoCoreError as e:
        print(f"Encountered an error while listing policies: {e}")


if __name__ == "__main__":
    main()
```

- Consulte detalhes da API em ListPolicies, na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
require 'aws-sdk-iam'
require 'logger'

# IAMManager is a class responsible for managing IAM operations
# such as listing all IAM policies in the current AWS account.
class IAMManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

De src/bin/hello.rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), Box<SdkError<ListPoliciesError>>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

De src/iam-service-lib.rs.

```
pub async fn list_policies(
```

```
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK for Rust.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conhecer os conceitos básicos do IAM com um AWS SDK

Os exemplos de código a seguir mostram como criar um usuário e assumir um perfil.

> **Warning**
>
> Para evitar riscos de segurança, não use usuários do IAM para autenticação ao desenvolver software com propósito específico ou trabalhar com dados reais. Em vez disso, use federação com um provedor de identidade, como Centro de Identidade do AWS IAM.

- Crie um usuário sem permissões.
- Crie uma função que conceda permissão para listar os buckets do Amazon S3 para a conta.
- Adicione uma política para permitir que o usuário assuma a função.
- Assuma o perfil e liste buckets do S3 usando credenciais temporárias, depois limpe os recursos.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;
```

```
    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });
```

```
        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
```

```
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
```

```
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
 userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
 DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
 policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
```

```
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
{ PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
```

```
   public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
   {
       var attachedPolicies = new List<AttachedPolicyType>();
       var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

       await foreach (var response in attachedRolePoliciesPaginator.Responses)
       {
           attachedPolicies.AddRange(response.AttachedPolicies);
       }

       return attachedPolicies;
   }


   /// <summary>
   /// List IAM groups.
   /// </summary>
   /// <returns>A list of IAM groups.</returns>
   public async Task<List<Group>> ListGroupsAsync()
   {
       var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
       var groups = new List<Group>();

       await foreach (var response in groupsPaginator.Responses)
       {
           groups.AddRange(response.Groups);
       }

       return groups;
   }


   /// <summary>
   /// List IAM policies.
   /// </summary>
   /// <returns>A list of the IAM policies.</returns>
   public async Task<List<ManagedPolicy>> ListPoliciesAsync()
   {
       var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
```

```
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
```

```
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
```

```
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
```

```
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
```

```
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
```

```
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
```

```
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);
```

```
        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);
```

```
        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
```

```
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();
```

```
            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
```

```
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
(IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
```

```
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);
```

```
        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para .NET.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
```

```
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
```

```
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
```

```
 fi

 local assume_role_policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"sts:AssumeRole\",
                  \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
```

```
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

As funções do IAM usadas neste cenário.

```
##############################################################################
# function iam_user_exists
#
```

```
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
```

```
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) key pair."
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
```

```
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }
```

```
# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
  return 1
fi
```

```
  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
```

```
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
```

```
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi

iecho "delete-access-key response:$response"
iecho
```

```
  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;
```

```
    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }
```

```
    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
```

```
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
```

```
                        listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
```

```
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
```

```
   bool result = true;
   if (policy.ArnHasBeenSet()) {
       // Detach the policy from the role.
       {
           Aws::IAM::Model::DetachRolePolicyRequest request;
           request.SetPolicyArn(policy.GetArn());
           request.SetRoleName(role.GetRoleName());

           Aws::IAM::Model::DetachRolePolicyOutcome outcome =
client.DetachRolePolicy(
                   request);
           if (!outcome.IsSuccess()) {
               std::cerr << "Error Detaching policy from roles. " <<
                         outcome.GetError().GetMessage() << std::endl;
               result = false;
           }
           else {
               std::cout << "Successfully detached the policy with arn "
                         << policy.GetArn()
                         << " from role " << role.GetRoleName() << "." <<
std::endl;
           }
       }

       // Delete the policy.
       {
           Aws::IAM::Model::DeletePolicyRequest request;
           request.WithPolicyArn(policy.GetArn());

           Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
           if (!outcome.IsSuccess()) {
               std::cerr << "Error deleting policy. " <<
                         outcome.GetError().GetMessage() << std::endl;
               result = false;
           }
           else {
               std::cout << "Successfully deleted the policy with arn "
                         << policy.GetArn() << std::endl;
           }
       }

   }
```

```
    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para C++.
  - AttachRolePolicy
  - CreateAccessKey

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "math/rand"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/credentials"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
```

```
 "github.com/aws/aws-sdk-go-v2/service/s3"
 "github.com/aws/aws-sdk-go-v2/service/sts"
 "github.com/aws/smithy-go"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/iam/actions"
)

// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
//  1. Create a user who has no permissions.
//  2. Create a role that grants permission to list Amazon Simple Storage Service
//     (Amazon S3) buckets for the account.
//  3. Add a policy to let the user assume the role.
//  4. Try and fail to list buckets without permissions.
//  5. Assume the role and list S3 buckets using temporary credentials.
//  6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig      aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper  actions.PolicyWrapper
 roleWrapper    actions.RoleWrapper
 userWrapper    actions.UserWrapper
 questioner     demotools.IQuestioner
 helper         IScenarioHelper
 isTestRun      bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
 helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:      sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
```

```
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser(ctx)
 accessKey := scenario.CreateAccessKey(ctx, user)
 role := scenario.CreateRoleAndPolicies(ctx, user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(ctx, accessKey)
 scenario.ListBucketsWithAssumedRole(ctx, noPermsConfig, role)
 scenario.Cleanup(ctx, user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
```

```
func (scenario AssumeRoleScenario) CreateUser(ctx context.Context) *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(ctx, userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(ctx, userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(ctx context.Context, user
 *types.User) *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(ctx, *user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(ctx context.Context,
 user *types.User) *types.Role {
```

```
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err := scenario.roleWrapper.CreateRole(ctx,
 scenario.helper.GetName(), *user.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  ctx, scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"},
 "arn:aws:s3:::*")
 if err != nil {
  panic(err)
 }
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(ctx, *listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {
  panic(err)
 }
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(ctx, *user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {
  panic(err)
 }
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
```

```
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(ctx
 context.Context, accessKey *types.AccessKey) *aws.Config {
 log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 noPermsConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
  ))
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

 s3Client := s3.NewFromConfig(noPermsConfig)
 _, err = s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
 if err != nil {
  // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
 } else {
  log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
   "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
 return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
```

```
//
//  1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//     the user's access key credentials.
//  2. Gets temporary credentials by assuming the role that grants permission to
 list the
//     buckets.
//  3. Creates an Amazon S3 client from the temporary credentials.
//  4. Lists buckets for the account. Because the temporary credentials are
 generated by
//     assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(ctx
 context.Context, noPermsConfig *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(ctx, &sts.AssumeRoleInput{
  RoleArn:         role.Arn,
  RoleSessionName: aws.String("AssumeRoleExampleSession"),
  DurationSeconds: aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(ctx,
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {
  panic(err)
 }

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(ctx, &s3.ListBucketsInput{})
```

```
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(ctx context.Context, user *types.User,
 role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
  policies, err := scenario.roleWrapper.ListAttachedRolePolicies(ctx,
 *role.RoleName)
  if err != nil {
   panic(err)
  }
  for _, policy := range policies {
   err = scenario.roleWrapper.DetachRolePolicy(ctx, *role.RoleName,
 *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   err = scenario.policyWrapper.DeletePolicy(ctx, *policy.PolicyArn)
   if err != nil {
    panic(err)
   }
   log.Printf("Detached policy %v from role %v and deleted the policy.\n",
    *policy.PolicyName, *role.RoleName)
  }
  err = scenario.roleWrapper.DeleteRole(ctx, *role.RoleName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted role %v.\n", *role.RoleName)

  userPols, err := scenario.userWrapper.ListUserPolicies(ctx, *user.UserName)
```

```
  if err != nil {
   panic(err)
  }
  for _, userPol := range userPols {
   err = scenario.userWrapper.DeleteUserPolicy(ctx, *user.UserName, userPol)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
  }
  keys, err := scenario.userWrapper.ListAccessKeys(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  for _, key := range keys {
   err = scenario.userWrapper.DeleteAccessKey(ctx, *user.UserName,
 *key.AccessKeyId)
   if err != nil {
    panic(err)
   }
   log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
  }
  err = scenario.userWrapper.DeleteUser(ctx, *user.UserName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleted user %v.\n", *user.UserName)
  log.Println(strings.Repeat("-", 88))
 }

}

// IScenarioHelper abstracts input and wait functions from a scenario so that
 they
// can be mocked for unit testing.
type IScenarioHelper interface {
 GetName() string
 Pause(secs int)
}

const rMax = 100000

type ScenarioHelper struct {
```

```
 Prefix string
 Random *rand.Rand
}

// GetName returns a unique name formed of a prefix and a random number.
func (helper *ScenarioHelper) GetName() string {
 return fmt.Sprintf("%v%v", helper.Prefix, helper.Random.Intn(rMax))
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 time.Sleep(time.Duration(secs) * time.Second)
}
```

Defina um struct que encapsule as ações de conta.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
```

```
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Defina um struct que encapsule as ações de política.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
```

```
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
```

```
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Defina um struct que encapsule as ações de perfil.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
```

```
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

```
// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

```
// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Defina um struct que encapsule as ações de usuário.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
```

```
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
```

```
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

```
// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
```

```
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Go.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações do usuário do IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the AWS IAM example scenario.");
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();
```

```
            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }
```

```
            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();
```

```
            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);
```

```
            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
```

```
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário e um perfil do IAM que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função. Após assumir a função, use credenciais temporárias para listar os buckets para a conta.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "iam_basic_test_username";
const policyName = "iam_basic_test_policy";
const roleName = "iam_basic_test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
```

```
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    }
    throw caught;
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
```

```
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
```

```
              Principal: {
                // Allow the previously created user to assume this role.
                AWS: User.Arn,
              },
              Action: "sts:AssumeRole",
            },
          ],
        }),
        RoleName: roleName,
      }),
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
```

```
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 120 }, () =>
  listBuckets(s3Client),
);
```

```
  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }
```

```
  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para JavaScript.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações do usuário do IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
```

```
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
<bucketName>

   Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
operation.
        fileLocation - The file location to the JSON required to create the role
(see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
read.
   """

   if (args.size != 6) {
        println(usage)
        exitProcess(1)
   }

   val userName = args[0]
   val policyName = args[1]
   val roleName = args[2]
   val roleSessionName = args[3]
   val fileLocation = args[4]
   val bucketName = args[5]

   createUser(userName)
   println("$userName was successfully created.")

   val polArn = createPolicy(policyName)
   println("The policy $polArn was successfully created.")

   val roleArn = createRole(roleName, fileLocation)
   println("$roleArn was successfully created.")
   attachRolePolicy(roleName, polArn)

   println("*** Wait for 1 MIN so the resource is available.")
   delay(60000)
   assumeGivenRole(roleArn, roleSessionName, bucketName)

   println("*** Getting ready to delete the AWS resources.")
   deleteRole(roleName, polArn)
   deleteUser(userName)
```

```
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:*"
                ],
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
```

```
    rolenameVal: String?,
    fileLocation: String?,
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
```

```
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String,
) {
    val stsClient = StsClient.fromEnvironment { region = "us-east-1" }
    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
```

```
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client.fromEnvironment {
        region = "us-east-1"
        credentialsProvider = staticCredentials
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String,
) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }
```

```
    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient.fromEnvironment { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey

- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
```

```
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
```

```
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para PHP.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário e um perfil do IAM que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função. Após assumir a função, use credenciais temporárias para listar os buckets para a conta.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
```

```
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
```

```
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
```

```
            f"Couldn't create a policy and attach it to role {role.name}. Here's
 why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
```

```
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
```

```
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
```

```
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
```

```
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário e um perfil do IAM que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função. Após assumir a função, use credenciais temporárias para listar os buckets para a conta.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts('Give AWS time to propagate resources...')
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info('Tried and failed to create demo user.')
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
 user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Principal: { 'AWS': user.arn },
        Action: 'sts:AssumeRole'
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
```

```
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 's3:ListAllMyBuckets',
        Resource: 'arn:aws:s3:::*'
      }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: '2012-10-17',
      Statement: [{
        Effect: 'Allow',
        Action: 'sts:AssumeRole',
        Resource: role.arn
      }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
```

```
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
  #
  # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
  def create_s3_resource(credentials)
    Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
  end

  # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
  # Because the resource uses credentials with limited access, it may not be able
to
  # list the S3 buckets.
  #
  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def list_buckets(s3_resource)
    count = 10
    s3_resource.buckets.each do |bucket|
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == 'AccessDenied'
      puts('Attempt to list buckets with no permissions: AccessDenied.')
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
```

```
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: 'create-use-assume-role-scenario'
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts('-' * 88)
  puts('Welcome to the IAM create a user and assume a role demo!')
  puts('-' * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts('Try to list buckets with credentials for a user who has no permissions.')
  puts('Expect AccessDenied from this call.')
  scenario.list_buckets(
```

```
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key))
  )
  puts('Now, assume the role that grants permission.')
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key)
  )
  puts('Here are your buckets:')
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts('Thanks for watching!')
  puts('-' * 88)
rescue Aws::Errors::ServiceError => e
  puts('Something went wrong with the demo.')
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Ruby.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}
```

```
async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
```

```
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
```

```
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(format!("iam_demo_assumerole_session_{uuid}"))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
```

```
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
```

```
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Ações do IAM usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do IAM com AWS SDKs. Cada exemplo inclui um link para o GitHub, em que é possível encontrar instruções para configurar e executar o código.

Esses trechos chamam a API do IAM e são trechos de código de programas maiores que devem ser executados no contexto. É possível ver as ações em contexto em Cenários para o IAM usando AWS SDKs.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a Referência de APIs do AWS Identity and Access Management.

Exemplos

- Usar AddClientIdToOpenIdConnectProvider com uma CLI
- Usar AddRoleToInstanceProfile com uma CLI
- Usar AddUserToGroup com uma CLI
- Usar AttachGroupPolicy com uma CLI
- Usar AttachRolePolicy com o AWS SDK ou a CLI
- Usar AttachUserPolicy com o AWS SDK ou a CLI
- Usar ChangePassword com uma CLI
- Usar CreateAccessKey com o AWS SDK ou a CLI
- Usar CreateAccountAlias com o AWS SDK ou a CLI
- Usar CreateGroup com o AWS SDK ou a CLI
- Usar CreateInstanceProfile com o AWS SDK ou a CLI
- Usar CreateLoginProfile com uma CLI
- Usar CreateOpenIdConnectProvider com uma CLI
- Usar CreatePolicy com o AWS SDK ou a CLI
- Usar CreatePolicyVersion com o AWS SDK ou a CLI
- Usar CreateRole com o AWS SDK ou a CLI
- Usar CreateSAMLProvider com o AWS SDK ou a CLI
- Usar CreateServiceLinkedRole com o AWS SDK ou a CLI
- Usar CreateUser com o AWS SDK ou a CLI

- Usar CreateVirtualMfaDevice com uma CLI
- Usar DeactivateMfaDevice com uma CLI
- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com uma CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com uma CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com uma CLI
- Usar DeleteOpenIdConnectProvider com uma CLI
- Usar DeletePolicy com o AWS SDK ou a CLI
- Usar DeletePolicyVersion com o AWS SDK ou a CLI
- Usar DeleteRole com o AWS SDK ou a CLI
- Usar DeleteRolePermissionsBoundary com uma CLI
- Usar DeleteRolePolicy com o AWS SDK ou a CLI
- Usar DeleteSAMLProvider com o AWS SDK ou a CLI
- Usar DeleteServerCertificate com o AWS SDK ou a CLI
- Usar DeleteServiceLinkedRole com o AWS SDK ou a CLI
- Usar DeleteSigningCertificate com uma CLI
- Usar DeleteUser com o AWS SDK ou a CLI
- Usar DeleteUserPermissionsBoundary com uma CLI
- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com uma CLI
- Usar DetachGroupPolicy com uma CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI
- Usar EnableMfaDevice com uma CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI
- Usar GenerateServiceLastAccessedDetails com uma CLI
- Usar GetAccessKeyLastUsed com o AWS SDK ou a CLI

- Usar GetAccountAuthorizationDetails com o AWS SDK ou a CLI
- Usar GetAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar GetAccountSummary com o AWS SDK ou a CLI
- Usar GetContextKeysForCustomPolicy com uma CLI
- Usar GetContextKeysForPrincipalPolicy com uma CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com uma CLI
- Usar GetGroupPolicy com uma CLI
- Usar GetInstanceProfile com uma CLI
- Usar GetLoginProfile com uma CLI
- Usar GetOpenIdConnectProvider com uma CLI
- Usar GetPolicy com o AWS SDK ou a CLI
- Usar GetPolicyVersion com o AWS SDK ou a CLI
- Usar GetRole com o AWS SDK ou a CLI
- Usar GetRolePolicy com uma CLI
- Usar GetSamlProvider com uma CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com uma CLI
- Usar GetServiceLastAccessedDetailsWithEntities com uma CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI
- Usar GetUser com o AWS SDK ou a CLI
- Usar GetUserPolicy com uma CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com uma CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI
- Usar ListAttachedUserPolicies com uma CLI
- Usar ListEntitiesForPolicy com uma CLI
- Usar ListGroupPolicies com uma CLI
- Usar ListGroups com o AWS SDK ou a CLI

- Usar ListGroupsForUser com uma CLI
- Usar ListInstanceProfiles com uma CLI
- Usar ListInstanceProfilesForRole com uma CLI
- Usar ListMfaDevices com uma CLI
- Usar ListOpenIdConnectProviders com uma CLI
- Usar ListPolicies com o AWS SDK ou a CLI
- Usar ListPolicyVersions com o AWS SDK ou a CLI
- Usar ListRolePolicies com o AWS SDK ou a CLI
- Usar ListRoleTags com uma CLI
- Usar ListRoles com o AWS SDK ou a CLI
- Usar ListSAMLProviders com o AWS SDK ou a CLI
- Usar ListServerCertificates com o AWS SDK ou a CLI
- Usar ListSigningCertificates com uma CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com uma CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com uma CLI
- Usar PutGroupPolicy com uma CLI
- Usar PutRolePermissionsBoundary com uma CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com uma CLI
- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com uma CLI
- Usar RemoveRoleFromInstanceProfile com uma CLI
- Usar RemoveUserFromGroup com uma CLI
- Usar ResyncMfaDevice com uma CLI
- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com uma CLI
- Usar TagUser com uma CLI
- Usar UntagRole com uma CLI

- Usar UntagUser com uma CLI
- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com uma CLI
- Usar UpdateAssumeRolePolicy com uma CLI
- Usar UpdateGroup com uma CLI
- Usar UpdateLoginProfile com uma CLI
- Usar UpdateOpenIdConnectProviderThumbprint com uma CLI
- Usar UpdateRole com uma CLI
- Usar UpdateRoleDescription com uma CLI
- Usar UpdateSamlProvider com uma CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com uma CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com uma CLI

Usar **`AddClientIdToOpenIdConnectProvider`** com uma CLI

Os exemplos de código a seguir mostram como usar o `AddClientIdToOpenIdConnectProvider`.

CLI

AWS CLI

Adicionar um ID de cliente (público) a um provedor Open-ID Connect (OIDC)

O comando `add-client-id-to-open-id-connect-provider` a seguir adiciona o ID do cliente `my-application-ID` ao provedor OIDC denominado `server.example.com`.

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Este comando não produz saída.

Para criar um provedor OIDC, use o comando `create-open-id-connect-provider`.

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AddClientIdToOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando adiciona o ID do cliente (ou público) **`my-application-ID`** ao provedor OIDC existente denominado **`server.example.com`**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Consulte detalhes da API em AddClientIdToOpenIdConnectProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando adiciona o ID do cliente (ou público) **`my-application-ID`** ao provedor OIDC existente denominado **`server.example.com`**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Para obter detalhes sobre a API, consulte AddClientIdToOpenIdConnectProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AddRoleToInstanceProfile`** com uma CLI

Os exemplos de código a seguir mostram como usar o `AddRoleToInstanceProfile`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Conceitos básicos de streaming gerenciado
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

CLI

AWS CLI

Adicionar um perfil a um perfil de instância

O comando `add-role-to-instance-profile` a seguir adiciona o perfil denominado `S3Access` ao perfil de instância denominado `Webserver`.

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

Este comando não produz saída.

Para criar um perfil de instância, use o comando `create-instance-profile`.

Para obter mais informações, consulte Uso de um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2 no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AddRoleToInstanceProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando adiciona o perfil denominado **`S3Access`** a um perfil de instância existente denominado **`webserver`**. Para criar o perfil de instância, use o comando **`New-IAMInstanceProfile`**. Depois de criar o perfil de instância e associá-lo a um perfil usando esse comando, você pode anexá-lo a uma instância do EC2. Para isso, use o cmdlet **`New-EC2Instance`** com o parâmetro **`InstanceProfile_Arn`** ou **`InstanceProfile-Name`** para executar a nova instância.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Consulte detalhes da API em AddRoleToInstanceProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando adiciona o perfil denominado **`S3Access`** a um perfil de instância existente denominado **`webserver`**. Para criar o perfil de instância, use o comando **`New-IAMInstanceProfile`**. Depois de criar o perfil de instância e associá-lo a um perfil usando esse comando, você pode anexá-lo a uma instância do EC2. Para isso, use o cmdlet **`New-EC2Instance`** com o parâmetro **`InstanceProfile_Arn`** ou **`InstanceProfile-Name`** para executar a nova instância.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Para obter detalhes sobre a API, consulte AddRoleToInstanceProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AddUserToGroup`** com uma CLI

Os exemplos de código a seguir mostram como usar o `AddUserToGroup`.

CLI

AWS CLI

Como adicionar um usuário a um grupo do IAM

O comando `add-user-to-group`, apresentado a seguir, adiciona um usuário do IAM denominado `Bob` ao grupo do IAM denominado `Admins`.

```
aws iam add-user-to-group \
    --user-name Bob \
```

```
    --group-name Admins
```

Este comando não produz saída.

Para obter mais informações, consulte Adicionar e remover usuários de um grupo de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AddUserToGroup na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando adiciona o usuário chamado **Bob** ao grupo denominado **Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Consulte detalhes da API em AddUserToGroup na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando adiciona o usuário chamado **Bob** ao grupo denominado **Admins**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Para obter detalhes sobre a API, consulte AddUserToGroup na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **AttachGroupPolicy** com uma CLI

Os exemplos de código a seguir mostram como usar o `AttachGroupPolicy`.

CLI

AWS CLI

Anexar uma política gerenciada a um grupo do IAM

O comando `attach-group-policy` a seguir anexa a política gerenciada da AWS denominada `ReadOnlyAccess` ao grupo do IAM denominado `Finance`.

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas gerenciadas e em linha no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AttachGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo anexa a política gerenciada pelo cliente denominada **`TesterPolicy`** ao grupo do IAM **`Testers`**. Os usuários desse grupo são imediatamente afetados pelas permissões definidas na versão padrão dessa política.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemplo 2: este exemplo anexa a política gerenciada AWS denominada **`AdministratorAccess`** ao grupo do IAM **`Admins`**. Os usuários desse grupo são imediatamente afetados pelas permissões definidas na versão mais recente dessa política.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Consulte detalhes da API em AttachGroupPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo anexa a política gerenciada pelo cliente denominada **`TesterPolicy`** ao grupo do IAM **`Testers`**. Os usuários desse grupo são imediatamente afetados pelas permissões definidas na versão padrão dessa política.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemplo 2: este exemplo anexa a política gerenciada AWS denominada **`AdministratorAccess`** ao grupo do IAM **`Admins`**. Os usuários desse grupo são imediatamente afetados pelas permissões definidas na versão mais recente dessa política.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Para obter detalhes sobre a API, consulte AttachGroupPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AttachRolePolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `AttachRolePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Configurar a conectividade do serviço de contêiner
- Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
- Criar uma API REST com integração de proxy de função
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Como criar sua primeira função com tecnologia sem servidor
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes

- Conceitos básicos de streaming gerenciado
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar funções
- Como mover segredos codificados para o armazenamento seguro de segredos
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor
- Trabalho com os fluxos e com a vida útil

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
```

```
    }
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # bashsupport disable=BP5008
 function usage() {
   echo "function iam_attach_role_policy"
   echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
   echo "  -n role_name   The name of the IAM role."
   echo "  -p policy_ARN -- The IAM policy document ARN."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "n:p:h" option; do
   case "${option}" in
     n) role_name="${OPTARG}" ;;
     p) policy_arn="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$role_name" ]]; then
   errecho "ERROR: You must provide a role name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$policy_arn" ]]; then
   errecho "ERROR: You must provide a policy ARN with the -p parameter."
   usage
   return 1
 fi

 response=$(aws iam attach-role-policy \
   --role-name "$role_name" \
   --policy-arn "$policy_arn")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em AttachRolePolicy na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
```

```
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
 iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
 <<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como anexar uma política gerenciada a um perfil do IAM

O comando `attach-role-policy`, apresentado a seguir, anexa a política gerenciada pela AWS denominada `ReadOnlyAccess` ao perfil do IAM denominado `ReadOnlyRole`.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas gerenciadas e em linha no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AttachRolePolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
```

```
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(ctx context.Context, policyArn
 string, roleName string) error {
 _, err := wrapper.IamClient.AttachRolePolicy(ctx, &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
```

```
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }
```

```
   public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
       try {
           ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                   .roleName(roleName)
                   .build();

           ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
           List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

           // Ensure that the policy is not attached to this role
           String polArn = "";
           for (AttachedPolicy policy : attachedPolicies) {
               polArn = policy.policyArn();
               if (polArn.compareTo(policyArn) == 0) {
                   System.out.println(roleName + " policy is already attached to
this role.");
                   return;
               }
           }

           AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                   .roleName(roleName)
                   .policyArn(policyArn)
                   .build();

           iam.attachRolePolicy(attachRequest);

           System.out.println("Successfully attached policy " + policyArn +
                   " to role " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       System.out.println("Done");
   }
}
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Anexe a política.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
```

```
      }
    });
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
```

```
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String,
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK for Kotlin.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
```

```
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo anexa a política gerenciada da AWS denominada **SecurityAudit** ao perfil do IAM **CoSecurityAuditors**. Os usuários que assumem esse perfil são imediatamente afetados pelas permissões definidas na versão mais recente dessa política.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Consulte detalhes da API em AttachRolePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo anexa a política gerenciada da AWS denominada **SecurityAudit** ao perfil do IAM **CoSecurityAuditors**. Os usuários que assumem esse perfil são imediatamente afetados pelas permissões definidas na versão mais recente dessa política.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Para obter detalhes sobre a API, consulte AttachRolePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Anexar uma política a uma função usando o objeto Policy do Boto3.

```
def attach_to_role(role_name, policy_arn):
```

```
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

Anexar uma política a uma função usando o objeto Role do Boto3.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, anexa e desconecta políticas de perfis.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
```

```
}
```

- Consulte detalhes da API em AttachRolePolicy na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    lo_iam->attachrolepolicy(
      iv_rolename = iv_role_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy attached to role successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Role or policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte AttachRolePolicy na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func attachRolePolicy(role: String, policyArn: String) async throws {
        let input = AttachRolePolicyInput(
            policyArn: policyArn,
            roleName: role
        )
        do {
            _ = try await client.attachRolePolicy(input: input)
        } catch {
            print("ERROR: Attaching a role policy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte AttachRolePolicy na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AttachUserPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `AttachUserPolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Criar usuários somente leitura e leitura/gravação usando

CLI

AWS CLI

Como anexar uma política gerenciada a um usuário do IAM

O comando `attach-user-policy`, apresentado a seguir, anexa a política gerenciada pela AWS denominada `AdministratorAccess` ao usuário do IAM denominado `Alice`.

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas gerenciadas e em linha no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AttachUserPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo anexa a política gerenciada da AWS denominada **`AmazonCognitoPowerUser`** ao usuário do IAM **`Bob`**. O usuário é imediatamente afetado pelas permissões definidas na versão mais recente dessa política.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Consulte detalhes da API em AttachUserPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo anexa a política gerenciada da AWS denominada **`AmazonCognitoPowerUser`** ao usuário do IAM **`Bob`**. O usuário é imediatamente afetado pelas permissões definidas na versão mais recente dessa política.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Para obter detalhes sobre a API, consulte AttachUserPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

- Consulte detalhes da API em AttachUserPolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  # Attaches a policy to a user
  #
  # @param user_name [String] The name of the user
  # @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_user(user_name, policy_arn)
    @iam_client.attach_user_policy(
      user_name: user_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to user: #{e.message}")
    false
  end
```

- Consulte detalhes da API em AttachUserPolicy na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;
```

```
    Ok(())
}
```

- Consulte detalhes da API em AttachUserPolicy na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    lo_iam->attachuserpolicy(
      iv_username = iv_user_name
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Policy attached to user successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'User or policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy attachment limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte AttachUserPolicy na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ChangePassword`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ChangePassword`.

CLI

AWS CLI

Alterar a senha do usuário do IAM

Para alterar a senha do usuário do IAM, recomendamos usar o parâmetro `--cli-input-json` para transmitir um arquivo JSON que contém suas senhas antigas e novas. Com esse método, você pode usar senhas fortes com caracteres não alfanuméricos. Pode ser difícil usar senhas com caracteres não alfanuméricos quando elas são transmitidas como parâmetros da linha de comando. Para usar o parâmetro `--cli-input-json`, comece usando o comando `change-password` com o parâmetro `--generate-cli-skeleton`, como no exemplo a seguir.

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

O comando anterior cria um arquivo JSON chamado change-password.json que você pode usar para preencher senhas antigas e novas. Por exemplo, o perfil pode ter a aparência a seguir.

```
{
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

Em seguida, para alterar a senha, use o comando `change-password` novamente, desta vez transmitindo o parâmetro `--cli-input-json` para especificar o arquivo JSON. O comando `change-password` a seguir usa o parâmetro `--cli-input-json` com um arquivo JSON chamado change-password.json.

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

Este comando não produz saída.

Esse comando pode ser chamado somente por usuários do IAM. Se esse comando for chamado usando credenciais da conta (raiz) da AWS, o comando retornará um erro `InvalidUserType`.

Para obter mais informações, consulte Como um usuário do IAM altera a própria senha no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ChangePassword na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando altera a senha do usuário que está executando o comando. Esse comando pode ser chamado somente por usuários do IAM. Se esse comando for chamado ao fazer login com as credenciais da conta (raiz) da AWS, o comando retornará um erro **`InvalidUserType`**.

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Consulte detalhes da API em ChangePassword na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando altera a senha do usuário que está executando o comando. Esse comando pode ser chamado somente por usuários do IAM. Se esse comando for chamado ao fazer login com as credenciais da conta (raiz) da AWS, o comando retornará um erro **`InvalidUserType`**.

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Para obter detalhes sobre a API, consulte ChangePassword na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateAccessKey`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateAccessKey`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Criar usuários somente leitura e leitura/gravação usando
- Gerenciar chaves de acesso

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
```

```
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
fi

if [[ -n "$file_name" ]]; then
  echo "$response" >"$file_name"
fi
```

```
  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Consulte detalhes da API em CreateAccessKey na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
```

```
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como criar uma chave de acesso para um usuário do IAM

O comando `create-access-key`, apresentado a seguir, cria uma chave de acesso (ID da chave de acesso e chave de acesso secreta) para o usuário do IAM denominado Bob.

```
aws iam create-access-key \
    --user-name Bob
```

Resultado:

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Armazene a chave de acesso secreta em um local seguro. Se ela for perdida, não será possível recuperá-la e você deverá criar uma nova chave de acesso.

Para obter mais informações, consulte Gerenciamento de chaves de acesso de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateAccessKey na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(ctx context.Context, userName
 string) (*types.AccessKey, error) {
 var key *types.AccessKey
```

```
 result, err := wrapper.IamClient.CreateAccessKey(ctx, &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
```

```
        final String usage = """

                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }

    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a chave de acesso.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
```

```
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma chave de acesso e um par de chaves de acesso secreto e os atribui ao usuário **David**. Certifique-se de salvar os valores **AccessKeyId** e **SecretAccessKey** em um arquivo, pois este é o único momento em que você pode obter a **SecretAccessKey**. Não será possível recuperá-la depois. Caso perca a chave secreta, você deve criar um par de chaves de acesso.

```
New-IAMAccessKey -UserName David
```

Saída:

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Para ver detalhes da API, consulte CreateAccessKey na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma chave de acesso e um par de chaves de acesso secreto e os atribui ao usuário **David**. Certifique-se de salvar os valores **AccessKeyId** e **SecretAccessKey** em um arquivo, pois este é o único momento em que você pode obter a **SecretAccessKey**. Não será possível recuperá-la depois. Caso perca a chave secreta, você deve criar um par de chaves de acesso.

```
New-IAMAccessKey -UserName David
```

Saída:

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Para obter detalhes sobre a API, consulte CreateAccessKey na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, desativa e exclui chaves de acesso.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
```

```
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
```

```
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em CreateAccessKey na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
```

```
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Consulte detalhes da API em CreateAccessKey na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->createaccesskey(
          iv_username = iv_user_name ).
        MESSAGE 'Access key created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Maximum number of access keys reached.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func createAccessKey(userName: String) async throws ->
 IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            print("ERROR: createAccessKey:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateAccountAlias`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateAccountAlias`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como criar um alias da conta

O comando `create-account-alias`, apresentado a seguir, cria o alias `examplecorp` para sua conta da AWS.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Este comando não produz saída.

Para obter mais informações, consulte O ID da sua conta da AWS e seu alias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateAccountAlias na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Criar o alias da conta.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.

- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo altera o alias da conta da AWS para **`mycompanyaws`**. O endereço da página de login do usuário muda para https://mycompanyaws.signin.aws.amazon.com/console. O URL original usando o número de ID da conta em vez do alias (https://<accountidnumber>.signin.aws.amazon.com/console) continua funcionando. No entanto, todos os URLs baseados em alias definidos anteriormente param de funcionar.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Para ver detalhes da API, consulte CreateAccountAlias na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo altera o alias da conta da AWS para **`mycompanyaws`**. O endereço da página de login do usuário muda para https://mycompanyaws.signin.aws.amazon.com/console. O URL original usando o número de ID da conta em vez do alias (https://<accountidnumber>.signin.aws.amazon.com/console) continua funcionando. No entanto, todos os URLs baseados em alias definidos anteriormente param de funcionar.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Para obter detalhes sobre a API, consulte CreateAccountAlias na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Listar, criar e excluir aliases da conta.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end
```

```
  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em CreateAccountAlias na Referência da API AWS SDK para Ruby.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
```

```
        lo_iam->createaccountalias(
          iv_accountalias = iv_account_alias ).
        MESSAGE 'Account alias created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Account alias already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Account alias limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte CreateAccountAlias na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateGroup`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateGroup`.

CLI

AWS CLI

Para criar um grupo do IAM

O comando `create-group`, apresentado a seguir, cria um grupo do IAM denominado `Admins`.

```
aws iam create-group \
    --group-name Admins
```

Resultado:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
```

```
        "GroupName": "Admins"
    }
}
```

Para obter mais informações, consulte Criação de grupos de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateGroup na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Consulte detalhes da API em CreateGroup na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um grupo do IAM denominado **`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Para ver detalhes da API, consulte CreateGroup na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um grupo do IAM denominado **`Developers`**.

```
New-IAMGroup -GroupName Developers
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Para obter detalhes sobre a API, consulte CreateGroup na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateInstanceProfile`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateInstanceProfile`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Criar e gerenciar um serviço resiliente
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Conceitos básicos de streaming gerenciado
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
```

```
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                "\"Version\": \"2012-10-17\"," +
                                "\"Statement\": [{" +
                                     "\"Effect\": \"Allow\"," +
                                     "\"Principal\": {" +
                                     "\"Service\": [" +
                                         "\"ec2.amazonaws.com\"" +
                                     "]" +
                                     "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
```

```
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
```

```
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Consulte detalhes da API em CreateInstanceProfile na Referência de API do AWS SDK para .NET.

CLI

AWS CLI

Como criar um perfil de instância

O comando `create-instance-profile`, apresentado a seguir, cria um perfil de instância denominado `Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Resultado:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Para adicionar um perfil a um perfil de instância, use o comando `add-role-to-instance-profile`.

Para obter mais informações, consulte Uso de um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2 no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateInstanceProfile na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
```

```
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
```

- Consulte detalhes da API em CreateInstanceProfile na Referência de API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um perfil de instância do IAM denominado **ProfileForDevEC2Instance**. Você deve executar o comando **Add-IAMRoleToInstanceProfile** separadamente para associar o perfil de instância a um perfil do IAM existente que fornece permissões à instância. Por fim, anexe o perfil de instância a uma instância do EC2 ao executá-la. Para isso, use o cmdlet **New-EC2Instance** com o parâmetro **InstanceProfile_Arn** ou **InstanceProfile_Name**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Saída:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Para ver detalhes da API, consulte CreateInstanceProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um perfil de instância do IAM denominado **ProfileForDevEC2Instance**. Você deve executar o comando **Add-IAMRoleToInstanceProfile** separadamente para associar o perfil de instância a um perfil do IAM existente que fornece permissões à instância. Por fim, anexe o perfil de instância

a uma instância do EC2 ao executá-la. Para isso, use o cmdlet **`New-EC2Instance`** com o parâmetro **`InstanceProfile_Arn`** ou **`InstanceProfile_Name`**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Saída:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Para obter detalhes sobre a API, consulte CreateInstanceProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo cria uma política, um perfil e um perfil de instância e vincula uns aos outros.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
```

```
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_instance_profile(
        self,
        policy_file: str,
```

```
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
```

```
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn
```

- Para obter detalhes da API, consulte CreateInstanceProfile na Referência da API do AWS SDK para Python (Boto3).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateLoginProfile`** com uma CLI

Os exemplos de código a seguir mostram como usar o `CreateLoginProfile`.

CLI

AWS CLI

Criar uma senha para um usuário do IAM

Para criar uma senha de um usuário do IAM, recomendamos usar o parâmetro `--cli-input-json` para transmitir um arquivo JSON que contém a senha. Usando esse método, você pode criar uma senha forte com caracteres não alfanuméricos. Pode ser difícil criar uma senha com caracteres não alfanuméricos ao transmiti-la como parâmetro da linha de comando.

Para usar o parâmetro `--cli-input-json`, comece usando o comando `create-login-profile` com o parâmetro `--generate-cli-skeleton`, como no exemplo a seguir.

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

O comando anterior cria um arquivo JSON chamado create-login-profile.json que pode ser usado para preencher as informações de um comando `create-login-profile` subsequente. Por exemplo:

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

Em seguida, para criar uma senha de um usuário do IAM, use o comando `create-login-profile` novamente, desta vez transmitindo o parâmetro `--cli-input-json` a fim de especificar o arquivo JSON. O comando `create-login-profile` a seguir usa o parâmetro `--cli-input-json` com um arquivo JSON chamado create-login-profile.json.

```
aws iam create-login-profile \
    --cli-input-json file://create-login-profile.json
```

Resultado:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

Se a nova senha violar a política de senha da conta, o comando retornará um erro `PasswordPolicyViolation`.

Para alterar a senha de um usuário que já tem uma, use `update-login-profile`. Para definir uma política de senha da conta, use o comando `update-account-password-policy`.

Se a política de senha da conta permitir, os usuários do IAM poderão alterar suas próprias senhas usando o comando `change-password`.

Para obter mais informações, consulte Gerenciamento de senhas de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma senha (temporária) para o usuário do IAM chamado Bob e define a sinalização que exige que o usuário altere a senha na próxima vez que **Bob** fizer login.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Saída:

```
CreateDate                 PasswordResetRequired             UserName
----------                 ---------------------             --------
4/14/2015 12:26:30 PM      True                              Bob
```

- Para ver detalhes da API, consulte CreateLoginProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma senha (temporária) para o usuário do IAM chamado Bob e define a sinalização que exige que o usuário altere a senha na próxima vez que **Bob** fizer login.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Saída:

```
CreateDate                  PasswordResetRequired                UserName
----------                  ---------------------                --------
4/14/2015 12:26:30 PM       True                                 Bob
```

- Para obter detalhes sobre a API, consulte CreateLoginProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateOpenIdConnectProvider`** com uma CLI

Os exemplos de código a seguir mostram como usar o `CreateOpenIdConnectProvider`.

CLI

AWS CLI

Criar um provedor OpenID Connect (OIDC)

Para criar um provedor OpenID Connect (OIDC), recomendamos usar o parâmetro `--cli-input-json` para transmitir um arquivo JSON que contém os parâmetros necessários. Ao criar um provedor OIDC, você deve transmitir o URL do provedor, e o URL deve começar com `https://`. Pode ser difícil transmitir o URL como parâmetro de linha de comando, porque os caracteres de dois pontos (:) e barra (/) têm um significado especial em alguns ambientes de linha de comando. Usar o parâmetro `--cli-input-json` contorna essa limitação.

Para usar o parâmetro `--cli-input-json`, comece usando o comando `create-open-id-connect-provider` com o parâmetro `--generate-cli-skeleton`, como no exemplo a seguir.

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

O comando anterior cria um arquivo JSON chamado create-open-id-connect-provider.json que você pode usar para preencher as informações de um comando `create-open-id-connect-provider` subsequente. Por exemplo:

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

Se a descoberta do seu provedor OIDC e os endpoints do JWKS usarem certificados diferentes, inclua impressões digitais de ambos os endpoints no `ThumbprintList`.

Em seguida, para criar o provedor OpenID Connect (OIDC), use o comando `create-open-id-connect-provider` novamente, desta vez transmitindo o parâmetro `--cli-input-json` a fim de especificar o arquivo JSON. O comando `create-open-id-connect-provider` a seguir usa o parâmetro `--cli-input-json` com um arquivo JSON chamado create-open-id-connect-provider.json.

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

Resultado:

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

Para obter mais informações sobre provedores OIDC, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

Para obter mais informações sobre como conseguir impressões digitais de um provedor OIDC, consulte Obter a impressão digital para um provedor de identidade OpenID Connect no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um provedor OIDC do IAM associado ao serviço do provedor compatível com OIDC encontrado no URL **`https://example.oidcprovider.com`** e no ID do cliente **`my-testapp-1`**. O provedor OIDC fornece a impressão digital. Para autenticar a impressão digital, siga as etapas em http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Saída:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Consulte detalhes da API em CreateOpenIdConnectProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um provedor OIDC do IAM associado ao serviço do provedor compatível com OIDC encontrado no URL **`https://example.oidcprovider.com`** e no ID do cliente **`my-testapp-1`**. O provedor OIDC fornece a impressão digital. Para autenticar a impressão digital, siga as etapas em http://docs.aws.amazon.com/IAM/latest/UserGuide/identity-providers-oidc-obtain-thumbprint.html.

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Saída:

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Para obter detalhes sobre a API, consulte CreateOpenIdConnectProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreatePolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreatePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Criar usuários somente leitura e leitura/gravação usando
- Como criar um espaço de trabalho de monitoramento gerenciado
- Conceitos básicos de streaming gerenciado
- Conceitos básicos da orquestração de fluxo de trabalaho
- Políticas gerenciadas
- Configuração do controle de acesso por atributo
- Como configurar o gerenciamento de sistemas
- Trabalhar com a API IAM Policy Builder

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
```

```
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
```

```
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}
```

- Consulte detalhes da API em CreatePolicy na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
```

```
    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
```

```
                << "   ]"
                << "}";

    return stringStream.str();
}
```

- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK para C++.

CLI

AWS CLI

Exemplo 1: como criar uma política gerenciada pelo cliente

O comando apresentado a seguir cria uma política gerenciada pelo cliente denominada `my-policy`. O arquivo `policy.json` é um documento JSON na pasta atual que concede acesso somente leitura à pasta `shared` em um bucket do Amazon S3 denominado `amzn-s3-demo-bucket`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json
```

Conteúdo de policy.json:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/shared/*"
            ]
        }
    ]
}
```

Resultado:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

Para obter mais informações sobre como usar arquivos como entrada para parâmetros de string, consulte Especificar valores de parâmetro para a AWS CLI no Guia do usuário da AWS CLI.

Exemplo 2: como criar uma política gerenciada pelo cliente com uma descrição

O comando a seguir cria uma política gerenciada pelo cliente denominada `my-policy` com uma descrição imutável.

O arquivo `policy.json` é um documento JSON na pasta atual que concede acesso a todas as ações Put, List e Get para um bucket do Amazon S3 denominado `amzn-s3-demo-bucket`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for amzn-s3-demo-bucket"
```

Conteúdo de policy.json:

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
```

```
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::amzn-s3-demo-bucket"
             ]
        }
    ]
}
```

Resultado:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Para obter mais informações sobre as políticas baseadas em identidade, consulte Políticas baseadas em identidade e em recurso no Guia do usuário do AWS IAM.

Exemplo 3: para criar uma política gerenciada pelo cliente com tags

O comando apresentado a seguir cria uma política gerenciada pelo cliente, denominada `my-policy`, com etiquetas. Este exemplo usa o parâmetro `--tags` com as seguintes tags formatadas em JSON: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Como alternativa, o parâmetro `--tags` pode ser usado com tags no formato abreviado: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

O arquivo `policy.json` é um documento JSON na pasta atual que concede acesso a todas as ações Put, List e Get para um bucket do Amazon S3 denominado `amzn-s3-demo-bucket`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Conteúdo de policy.json:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Resultado:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
```

```
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {
        ]
    }
}
```

Para obter mais informações sobre as políticas de marcação, consulte Marcar políticas gerenciadas pelo cliente no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreatePolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
```

```
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version   string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect    string
 Action    []string
 Principal map[string]string `json:",omitempty"`
 Resource  *string           `json:",omitempty"`
}

// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(ctx context.Context, policyName string,
 actions []string,
 resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
```

```
 }
 result, err := wrapper.IamClient.CreatePolicy(ctx, &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
```

```
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
```

```
        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
 + result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);

            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a política.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.

- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
```

```
  PolicyName: "myDynamoDBPolicy",
};

iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal = """
    {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "dynamodb:DeleteItem",
                    "dynamodb:GetItem",
                    "dynamodb:PutItem",
                    "dynamodb:Scan",
                    "dynamodb:UpdateItem"
                ],
```

```
                "Resource": "*"
            }
        ]
    }
    """.trimIndent()

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Consulte detalhes da API em CreatePolicy na Referência da API AWS SDK for Kotlin.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
```

```
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    /**
     * @param string $policyName
     * @param string $policyDocument
     * @return array
     */
    public function createPolicy(string $policyName, string $policyDocument)
    {
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma política do IAM na conta atual da AWS denominada **MySamplePolicy**. O arquivo **MySamplePolicy.json** fornece o conteúdo da política. Observe que você deve usar o parâmetro switch **-Raw** para processar com êxito o arquivo de política JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Saída:

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
```

```
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Para ver detalhes da API, consulte CreatePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma política do IAM na conta atual da AWS denominada **`MySamplePolicy`**. O arquivo **`MySamplePolicy.json`** fornece o conteúdo da política. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o arquivo de política JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Saída:

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Para obter detalhes sobre a API, consulte CreatePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
```

```
        return policy
```

- Consulte detalhes da API em CreatePolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, anexa e desconecta políticas de perfis.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
```

```
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Para obter detalhes, consulte CreatePolicy na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
```

```
) -> Result<Policy, iamError> {
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
    Ok(policy.policy.unwrap())
}
```

- Consulte detalhes da API em CreatePolicy na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->createpolicy(
          iv_policyname = iv_policy_name
          iv_policydocument = iv_policy_document
          iv_description = iv_description ).
        MESSAGE 'Policy created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Policy already exists.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Policy limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte CreatePolicy na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: createPolicy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreatePolicy na Referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreatePolicyVersion`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreatePolicyVersion`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Políticas gerenciadas

CLI

AWS CLI

Como criar uma nova versão de uma política gerenciada

Este exemplo cria uma nova versão `v2` da política do IAM cujo ARN é `arn:aws:iam::123456789012:policy/MyPolicy` e a torna a versão padrão.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Resultado:

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

Para obter mais informações, consulte Versionamento de políticas do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreatePolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma versão "v2" da política do IAM cujo ARN é **`arn:aws:iam::123456789012:policy/MyPolicy`** e a torna a versão padrão. O arquivo **`NewPolicyVersion.json`** fornece o conteúdo da política. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o arquivo de política JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Saída:

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Para ver detalhes da API, consulte CreatePolicyVersion na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma versão "v2" da política do IAM cujo ARN é **`arn:aws:iam::123456789012:policy/MyPolicy`** e a torna a versão padrão. O arquivo **`NewPolicyVersion.json`** fornece o conteúdo da política. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o arquivo de política JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Saída:

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
```

```
4/15/2015 10:54:54 AM                                                  True
          v2
```

- Para obter detalhes sobre a API, consulte CreatePolicyVersion na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
```

```
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version
```

- Consulte detalhes da API em CreatePolicyVersion na Referência da API AWS SDK para Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->createpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_policydocument = iv_policy_document
      iv_setasdefault = iv_set_as_default ).
    MESSAGE 'Policy version created successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iammalformedplydocex.
    MESSAGE 'Policy document is malformed.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Policy version limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte CreatePolicyVersion na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateRole`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateRole`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Configurar a conectividade do serviço de contêiner
- Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
- Criar uma API REST com integração de proxy de função
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Como criar sua primeira função com tecnologia sem servidor
- Conceitos básicos de data warehouses de tecnologia sem servidor
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos de clusters de data warehouse provisionados
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar funções
- Como mover segredos codificados para o armazenamento seguro de segredos

- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor
- Trabalho com os fluxos e com a vida útil

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
```

```
  echo "  -n role_name   The name of the IAM role."
  echo "  -p policy_json -- The assume role policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em CreateRole na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
```

```
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK para C++.

CLI

AWS CLI

Exemplo 1: como criar um perfil do IAM

O comando `create-role`, apresentado a seguir, cria um perfil, denominado `Test-Role`, e anexa uma política de confiança a ele.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Resultado:

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

A política de confiança é definida como um documento JSON no arquivo Test-Role-Trust-Policy.json. (O nome e a extensão do arquivo não têm significado.) A política de confiança deve especificar uma entidade principal.

Para anexar uma política de permissões a um perfil, use o comando `put-role-policy`.

Para obter mais informações, consulte Criação de perfis do IAM no Guia do usuário do AWS IAM.

Exemplo 2: como criar um perfil do IAM com a duração máxima da sessão especificada

O comando `create-role`, apresentado a seguir, cria um perfil denominado `Test-Role` e define a duração máxima da sessão como 7.200 segundos (duas horas).

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

Resultado:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Para obter mais informações, consulte Modificar a duração máxima da sessão de um perfil (API da AWS) no Guia do usuário do AWS IAM.

Exemplo 3: como criar um perfil do IAM com etiquetas

O comando apresentado a seguir cria um perfil do IAM `Test-Role` com etiquetas. Este exemplo usa o sinalizador de parâmetro `--tags` com as seguintes etiquetas formatadas em JSON: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Como alternativa, o sinalizador `--tags` pode ser usado com etiquetas no formato abreviado: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Resultado:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
```

```
                "Value": "Seattle"
            }
        ]
    }
}
```

Para obter mais informações, consulte Marcar perfis do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
```

```
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(ctx context.Context, roleName string,
 trustedUserArn string) (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version: "2012-10-17",
  Statement: []PolicyStatement{{
   Effect:    "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action:    []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(ctx, &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s

                Where:
                    rolename - The name of the role to create.\s
```

```
                    fileLocation - The location of the JSON document that
represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
```

```
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a função.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
          },
          Action: "sts:AssumeRole",
        },
      ],
```

```
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Consulte detalhes da API em CreateRole na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
```

```
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um perfil denominado **MyNewRole** e anexa a ele a política encontrada no arquivo **NewRoleTrustPolicy.json**. Observe que você deve usar o parâmetro switch **-Raw** para processar com êxito o arquivo de política JSON. O documento de política exibido na saída é codificado em URL. Ele é decodificado nesse exemplo com o método .NET **UrlDecode**.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Saída:

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                             %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                             %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                             %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
```

```
                              %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                              %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                              %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Consulte detalhes da API em CreateRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um perfil denominado **`MyNewRole`** e anexa a ele a política encontrada no arquivo **`NewRoleTrustPolicy.json`**. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o arquivo de política JSON. O documento de política exibido na saída é codificado em URL. Ele é decodificado nesse exemplo com o método .NET **`UrlDecode`**.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Saída:

```
Arn                       : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument  : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
                            %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                            %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                            %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                            %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate                : 4/15/2015 11:04:23 AM
Path                      : /
RoleId                    : V5PAJI2KPN4EAEXAMPLE1
RoleName                  : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Para obter detalhes sobre a API, consulte CreateRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Creates a role and attaches policies to it.
 #
 # @param role_name [String] The name of the role.
 # @param assume_role_policy_document [Hash] The trust relationship policy
document.
 # @param policy_arns [Array<String>] The ARNs of the policies to attach.
 # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
 def create_role(role_name, assume_role_policy_document, policy_arns)
   response = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: assume_role_policy_document.to_json
   )
   role_arn = response.role.arn

   policy_arns.each do |policy_arn|
     @iam_client.attach_role_policy(
       role_name: role_name,
       policy_arn: policy_arn
     )
   end

   role_arn
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating role: #{e.message}")
   nil
```

```
  end
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Consulte detalhes da API em CreateRole na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->createrole(
          iv_rolename = iv_role_name
          iv_assumerolepolicydocument = iv_assume_role_policy_document ).
        MESSAGE 'Role created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'Role already exists.' TYPE 'E'.
      CATCH /aws1/cx_iammalformedplydocex.
        MESSAGE 'Assume role policy document is malformed.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
        MESSAGE 'Role limit exceeded.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte CreateRole na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3
```

```
   public func createRole(name: String, policyDocument: String) async throws ->
String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            print("ERROR: createRole:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateRole na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateSAMLProvider`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateSAMLProvider`.

CLI

AWS CLI

Como criar um provedor SAML

Este exemplo cria um novo provedor SAML no IAM denominado `MySAMLProvider`. Ele é descrito pelo documento de metadados do SAML, que encontra-se no arquivo `SAMLMetaData.xml`.

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

Resultado:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Para obter mais informações, consulte Criação de provedores de identidade SAML do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateSAMLProvider na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import * as path from "node:path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
```

```
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Consulte detalhes da API em CreateSAMLProvider na Referência da API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma entidade provedora SAML no IAM. Ele é denominado **`MySAMLProvider`** e descrito pelo documento de metadados SAML encontrado no arquivo **`SAMLMetaData.xml`**, que foi baixado separadamente do site do provedor de serviços SAML.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Saída:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Para ver detalhes da API, consulte CreateSAMLProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma entidade provedora SAML no IAM. Ele é denominado **`MySAMLProvider`** e descrito pelo documento de metadados SAML encontrado no arquivo **`SAMLMetaData.xml`**, que foi baixado separadamente do site do provedor de serviços SAML.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Saída:

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Para obter detalhes sobre a API, consulte CreateSAMLProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateServiceLinkedRole`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateServiceLinkedRole`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Create an IAM service-linked role.
```

```
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como criar um perfil vinculado ao serviço

O exemplo que usa `create-service-linked-role`, apresentado a seguir, cria um perfil vinculado ao serviço para o serviço da AWS especificado e anexa a descrição especificada.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Resultado:

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
```

```
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Para obter mais informações, consulte Usar perfis vinculados ao serviço no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateServiceLinkedRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(ctx context.Context,
 serviceName string, description string) (
 *types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(ctx,
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Criar uma função vinculada ao serviço.

```
import {
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
```

```
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    }
    throw caught;
  }
};
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
```

```
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um perfil vinculado ao serviço para o serviço de ajuste de escala automático.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Para ver detalhes da API, consulte CreateServiceLinkedRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um perfil vinculado ao serviço para o serviço de ajuste de escala automático.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Para obter detalhes sobre a API, consulte CreateServiceLinkedRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
        raise
    else:
        return role
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix
   )
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
   role_name
 rescue Aws::Errors::ServiceError => e
   @logger.error("Couldn't create service-linked role for #{service_name}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   raise
 end
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- Consulte detalhes da API em CreateServiceLinkedRole na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->listpolicyversions(
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func createServiceLinkedRole(service: String, suffix: String? = nil,
 description: String?)
                    async throws -> IAMClientTypes.Role {
        let input = CreateServiceLinkedRoleInput(
            awsServiceName: service,
            customSuffix: suffix,
            description: description
        )
        do {
            let output = try await client.createServiceLinkedRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
```

```
            return role
        } catch {
            print("ERROR: createServiceLinkedRole:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateUser`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `CreateUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Criar usuários somente leitura e leitura/gravação usando

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
/// <summary>
/// Create an IAM user.
/// </summary>
/// <param name="userName">The username for the new IAM user.</param>
/// <returns>The IAM user that was created.</returns>
```

```
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em CreateUser na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);

    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para C++.

CLI

AWS CLI

Exemplo 1: como criar um usuário do IAM

O comando `create-user`, apresentado a seguir, cria um usuário do IAM denominado Bob na conta atual.

```
aws iam create-user \
    --user-name Bob
```

Resultado:

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    }
}
```

Para obter mais informações, consulte Criar um usuário do IAM na sua conta da AWS no Guia do usuário do AWS IAM.

Exemplo 2: como criar um usuário do IAM em um caminho especificado

O comando `create-user`, apresentado a seguir, cria um usuário do IAM denominado `Bob` no caminho especificado.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Resultado:

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Para obter mais informações, consulte Identificadores do IAM no Guia do usuário do AWS IAM.

Exemplo 3: como criar um usuário do IAM com etiquetas

O comando `create-user`, apresentado a seguir, cria um usuário do IAM denominado Bob com etiquetas. Este exemplo usa o sinalizador de parâmetro `--tags` com as seguintes etiquetas formatadas em JSON: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Como alternativa, o sinalizador `--tags` pode ser usado com etiquetas no formato abreviado: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Resultado:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Para obter mais informações, consulte Marcar usuários do IAM no Guia do usuário do AWS IAM.

Exemplo 3: como criar um usuário do IAM com um limite de permissões definido

O comando `create-user`, apresentado a seguir, cria um usuário do IAM denominado Bob com o limite de permissões AmazonS3FullAccess.

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Resultado:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

Para obter mais informações, consulte Limites de permissões para entidades do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateUser na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(ctx, &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
```

```
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
 iamWaiter.waitUntilUserExists(userRequest);

 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie o usuário.

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
```

```
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK for Kotlin.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
```

```
        ]);

        return $result['User'];
    }
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um usuário do IAM chamado **Bob**. Se Bob precisar entrar no console da AWS, você deve executar o comando **New-IAMLoginProfile** separadamente para criar um perfil de login com uma senha. Se Bob precisar executar comandos do PowerShell da AWS ou da CLI entre plataformas ou fazer chamadas de API da AWS, você deve executar o comando **New-IAMAccessKey** separadamente para criar chaves de acesso.

```
New-IAMUser -UserName Bob
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Consulte detalhes da API em CreateUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um usuário do IAM chamado **Bob**. Se Bob precisar entrar no console da AWS, você deve executar o comando **New-IAMLoginProfile** separadamente para criar um perfil de login com uma senha. Se Bob precisar executar comandos do PowerShell da AWS ou da CLI entre plataformas ou fazer chamadas de API da AWS, você deve executar o comando **New-IAMAccessKey** separadamente para criar chaves de acesso.

```
New-IAMUser -UserName Bob
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Para obter detalhes sobre a API, consulte CreateUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Creates a user and their login profile
 #
 # @param user_name [String] The name of the user
 # @param initial_password [String] The initial password for the user
 # @return [String, nil] The ID of the user if created, or nil if an error
occurred
 def create_user(user_name, initial_password)
   response = @iam_client.create_user(user_name: user_name)
   @iam_client.wait_until(:user_exists, user_name: user_name)
   @iam_client.create_login_profile(
     user_name: user_name,
     password: initial_password,
     password_reset_required: true
   )
   @logger.info("User '#{user_name}' created successfully.")
   response.user.user_id
 rescue Aws::IAM::Errors::EntityAlreadyExists
   @logger.error("Error creating user '#{user_name}': user already exists.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating user '#{user_name}': #{e.message}")
   nil
 end
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;

    Ok(response.user.unwrap())
}
```

- Consulte detalhes da API em CreateUser na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->createuser(
          iv_username = iv_user_name ).
        MESSAGE 'User created successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'User already exists.' TYPE 'E'.
      CATCH /aws1/cx_iamlimitexceededex.
```

```
        MESSAGE 'Limit exceeded for IAM users.' TYPE 'E'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Entity does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte CreateUser na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func createUser(name: String) async throws -> String {
        let input = CreateUserInput(
            userName: name
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            print("ERROR: createUser:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateUser na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateVirtualMfaDevice`** com uma CLI

Os exemplos de código a seguir mostram como usar o `CreateVirtualMfaDevice`.

CLI

AWS CLI

Criar um dispositivo de MFA virtual

Este exemplo cria um dispositivo de MFA virtual denominado `BobsMFADevice`. Ele cria um arquivo contendo informações de bootstrap denominadas `QRCode.png` e as coloca no diretório `C:/`. O método de bootstrap usado neste exemplo é `QRCodePNG`.

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

Resultado:

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

Para obter mais informações, consulte Uso de autenticação multifator (MFA) na AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em CreateVirtualMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria um dispositivo de MFA virtual. As linhas 2 e 3 extraem o valor de **`Base32StringSeed`** de que o programa de software de MFA virtual precisa para criar uma conta (como alternativa ao código QR). Depois de configurar o programa com o valor, obtenha dois códigos de autenticação sequencial do programa. Por fim, use o último comando para vincular o dispositivo de MFA virtual ao usuário do IAM **`Bob`** e sincronizar a conta com os dois códigos de autenticação.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Saída:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Exemplo 2: este exemplo cria um dispositivo de MFA virtual. As linhas 2 e 3 extraem o valor de **`QRCodePNG`** e o gravam em um arquivo. Essa imagem pode ser digitalizada pelo programa de software de MFA virtual para criar uma conta (como alternativa à inserção manual do valor de Base32StringSeed). Depois de criar a conta no programa de MFA virtual, obtenha dois códigos de autenticação sequencial e insira-os nos últimos comandos para vincular o dispositivo MFA virtual ao usuário do IAM **`Bob`** e sincronizar a conta.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Saída:

```
 -- Pause here to scan PNG with virtual MFA program to register account. --
```

```
Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Para ver detalhes da API, consulte CreateVirtualMfaDevice na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria um dispositivo de MFA virtual. As linhas 2 e 3 extraem o valor de **`Base32StringSeed`** de que o programa de software de MFA virtual precisa para criar uma conta (como alternativa ao código QR). Depois de configurar o programa com o valor, obtenha dois códigos de autenticação sequencial do programa. Por fim, use o último comando para vincular o dispositivo de MFA virtual ao usuário do IAM **`Bob`** e sincronizar a conta com os dois códigos de autenticação.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Saída:

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Exemplo 2: este exemplo cria um dispositivo de MFA virtual. As linhas 2 e 3 extraem o valor de **`QRCodePNG`** e o gravam em um arquivo. Essa imagem pode ser digitalizada pelo programa de software de MFA virtual para criar uma conta (como alternativa à inserção manual do valor de Base32StringSeed). Depois de criar a conta no programa de MFA virtual, obtenha dois códigos de autenticação sequencial e insira-os nos últimos comandos para vincular o dispositivo MFA virtual ao usuário do IAM **`Bob`** e sincronizar a conta.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
```

```
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Saída:

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Para obter detalhes sobre a API, consulte CreateVirtualMfaDevice na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeactivateMfaDevice`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeactivateMfaDevice`.

CLI

AWS CLI

Desativar um dispositivo de MFA

Esse comando desativa o dispositivo de MFA virtual com o ARN `arn:aws:iam::210987654321:mfa/BobsMFADevice` associado ao usuário Bob.

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

Este comando não produz saída.

Para obter mais informações, consulte Uso de autenticação multifator (MFA) na AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeactivateMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando desabilita o dispositivo de MFA de hardware associado ao usuário **Bob** que tem o número de série **123456789012**.

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Exemplo 2: este comando desativa o dispositivo de MFA virtual associado ao usuário **David** que tem o ARN **arn:aws:iam::210987654321:mfa/David**. Observe que o dispositivo de MFA virtual não é excluído da conta. O dispositivo virtual ainda está presente e aparece na saída do comando **Get-IAMVirtualMFADevice**. Antes de criar um dispositivo de MFA virtual para o mesmo usuário, você deve excluir o antigo usando o comando **Remove-IAMVirtualMFADevice**.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Consulte detalhes da API em DeactivateMfaDevice na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando desabilita o dispositivo de MFA de hardware associado ao usuário **Bob** que tem o número de série **123456789012**.

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Exemplo 2: este comando desativa o dispositivo de MFA virtual associado ao usuário **David** que tem o ARN **arn:aws:iam::210987654321:mfa/David**. Observe que o dispositivo de MFA virtual não é excluído da conta. O dispositivo virtual ainda está presente e aparece na saída do comando **Get-IAMVirtualMFADevice**. Antes de criar um dispositivo de MFA virtual para o mesmo usuário, você deve excluir o antigo usando o comando **Remove-IAMVirtualMFADevice**.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Para obter detalhes sobre a API, consulte DeactivateMfaDevice na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteAccessKey`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteAccessKey`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Criar usuários somente leitura e leitura/gravação usando
- Gerenciar chaves de acesso

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
```

```
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an AWS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
```

```
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Consulte detalhes da API em DeleteAccessKey na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como excluir uma chave de acesso para um usuário do IAM

O comando `delete-access-key`, apresentado a seguir, exclui a chave de acesso especificada (ID da chave de acesso e chave de acesso secreta) para o usuário do IAM denominado Bob.

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Este comando não produz saída.

Para listar as chaves de acesso definidas para um usuário do IAM, use o comando `list-access-keys`.

Para obter mais informações, consulte Gerenciamento de chaves de acesso de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteAccessKey na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(ctx context.Context, userName string,
 keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(ctx, &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
```

```
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s
```

```
                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Exclua a chave de acesso.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String,
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o par de chaves de acesso AWS com o ID da chave **`AKIAIOSFODNN7EXAMPLE`** do usuário chamado **`Bob`**.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Consulte detalhes da API em DeleteAccessKey na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o par de chaves de acesso AWS com o ID da chave **`AKIAIOSFODNN7EXAMPLE`** do usuário chamado **`Bob`**.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Para obter detalhes sobre a API, consulte DeleteAccessKey na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, desativa e exclui chaves de acesso.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
```

```
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
```

```
    false
  end
end
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
        }
    }
    Ok(())
```

```
}
```

- Consulte detalhes da API em DeleteAccessKey na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    lo_iam->deleteaccesskey(
      iv_accesskeyid = iv_access_key_id
      iv_username = iv_user_name ).
    MESSAGE 'Access key deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Access key or user does not exist.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
```

```
import AWSS3


    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws
    {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }

        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            print("ERROR: deleteAccessKey:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API do AWS SDK para Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteAccountAlias`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteAccountAlias`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como excluir um alias da conta

O comando `delete-account-alias`, apresentado a seguir, remove o alias `mycompany` para a conta atual.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Este comando não produz saída.

Para obter mais informações, consulte O ID da sua conta da AWS e seu alias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteAccountAlias na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
```

```
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Exclua o alias da conta.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo remove o alias da conta da Conta da AWS. A página de login do usuário com o alias em https://mycompanyaws.signin.aws.amazon.com/console no longer works. Em vez disso, você deve usar o URL original com o número do ID da Conta da AWS em https://<accountidnumber>.signin.aws.amazon.com/console.

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Consulte detalhes da API em DeleteAccountAlias na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo remove o alias da conta da Conta da AWS. A página de login do usuário com o alias em https://mycompanyaws.signin.aws.amazon.com/console no longer works. Em vez disso, você deve usar o URL original com o número do ID da Conta da AWS em https://<accountidnumber>.signin.aws.amazon.com/console.

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Para obter detalhes sobre a API, consulte DeleteAccountAlias na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Listar, criar e excluir aliases da conta.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em DeleteAccountAlias na Referência da API AWS SDK para Ruby.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->deleteaccountalias(
          iv_accountalias = iv_account_alias ).
        MESSAGE 'Account alias deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Account alias does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte DeleteAccountAlias na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteAccountPasswordPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteAccountPasswordPolicy`.

CLI

AWS CLI

Excluir a política de senha da conta atual

O comando `delete-account-password-policy` a seguir remove a política de senha da conta atual.

```
aws iam delete-account-password-policy
```

Este comando não produz saída.

Para obter mais informações, consulte Definição de uma política de senhas de contas para usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteAccountPasswordPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui a política de senha da Conta da AWS e redefine todos os valores para os padrões originais. Se uma política de senha não existir no momento, a seguinte mensagem de erro será exibida: The account policy with name PasswordPolicy cannot be found

```
Remove-IAMAccountPasswordPolicy
```

- Consulte detalhes da API em DeleteAccountPasswordPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui a política de senha da Conta da AWS e redefine todos os valores para os padrões originais. Se uma política de senha não existir no momento, a seguinte mensagem de erro será exibida: The account policy with name PasswordPolicy cannot be found.

```
Remove-IAMAccountPasswordPolicy
```

- Para obter detalhes sobre a API, consulte DeleteAccountPasswordPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteGroup`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteGroup`.

CLI

AWS CLI

Como excluir um grupo do IAM

O comando `delete-group`, apresentado a seguir, exclui um grupo do IAM denominado `MyTestGroup`.

```
aws iam delete-group \
    --group-name MyTestGroup
```

Este comando não produz saída.

Para obter mais informações, consulte Exclusão de um grupo de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteGroup na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Consulte detalhes da API em DeleteGroup na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o grupo do IAM denominado **MyTestGroup**. O primeiro comando remove todos os usuários do IAM que são membros do grupo, e o segundo exclui o grupo do IAM. Ambos os comandos funcionam sem nenhuma solicitação de confirmação.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Consulte detalhes da API em DeleteGroup na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o grupo do IAM denominado **MyTestGroup**. O primeiro comando remove todos os usuários do IAM que são membros do grupo, e o segundo exclui o grupo do IAM. Ambos os comandos funcionam sem nenhuma solicitação de confirmação.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Para obter detalhes sobre a API, consulte DeleteGroup na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteGroupPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteGroupPolicy`.

CLI

AWS CLI

Como excluir uma política de um grupo do IAM

O comando `delete-group-policy`, apresentado a seguir, exclui a política denominada `ExamplePolicy` do grupo denominado `Admins`.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Este comando não produz saída.

Para visualizar as políticas anexadas a um grupo, use o comando `list-group-policies`.

Para obter mais informações, consulte Gerenciamento de políticas do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo remove a política em linha denominada **`TesterPolicy`** do grupo do IAM **`Testers`**. Os usuários desse grupo perdem imediatamente as permissões definidas nessa política.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Consulte detalhes da API em DeleteGroupPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo remove a política em linha denominada **`TesterPolicy`** do grupo do IAM **`Testers`**. Os usuários desse grupo perdem imediatamente as permissões definidas nessa política.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Para obter detalhes sobre a API, consulte DeleteGroupPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteInstanceProfile`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteInstanceProfile`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Criar e gerenciar um serviço resiliente
- Conceitos básicos de streaming gerenciado
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
   /// <summary>
   /// Detaches a role from an instance profile, detaches policies from the
role,
   /// and deletes all the resources.
   /// </summary>
   /// <param name="profileName">The name of the profile to delete.</param>
   /// <param name="roleName">The name of the role to delete.</param>
   /// <returns>Async task.</returns>
   public async Task DeleteInstanceProfile(string profileName, string roleName)
   {
       try
       {
           await _amazonIam.RemoveRoleFromInstanceProfileAsync(
               new RemoveRoleFromInstanceProfileRequest()
               {
                   InstanceProfileName = profileName,
                   RoleName = roleName
               });
           await _amazonIam.DeleteInstanceProfileAsync(
               new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
           var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
               new ListAttachedRolePoliciesRequest() { RoleName = roleName });
           foreach (var policy in attachedPolicies.AttachedPolicies)
           {
               await _amazonIam.DetachRolePolicyAsync(
                   new DetachRolePolicyRequest()
                   {
                       RoleName = roleName,
                       PolicyArn = policy.PolicyArn
                   });
```

```
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- Consulte detalhes da API em DeleteInstanceProfile na Referência da API do AWS SDK para .NET.

CLI

AWS CLI

Como excluir um perfil de instância

O comando `delete-instance-profile`, apresentado a seguir, exclui o perfil de instância denominado `ExampleInstanceProfile`.

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Este comando não produz saída.

Para obter mais informações, consulte Usar perfis de instância no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteInstanceProfile na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- Consulte detalhes da API em DeleteInstanceProfile na Referência da API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o perfil de instância do EC2 denominado **`MyAppInstanceProfile`**. O primeiro comando desassocia todos os perfis do perfil de instância e, em seguida, o segundo comando exclui o perfil de instância.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Consulte detalhes da API em DeleteInstanceProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o perfil de instância do EC2 denominado **`MyAppInstanceProfile`**. O primeiro comando desassocia todos os perfis do perfil de instância e, em seguida, o segundo comando exclui o perfil de instância.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Para obter detalhes sobre a API, consulte DeleteInstanceProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo remove o perfil do perfil de instância, separa todas as políticas anexadas ao perfil e exclui todos os recursos.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
```

```
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
```

```
self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
```

- Para obter detalhes da API, consulte DeleteInstanceProfile na Referência da API do AWS SDK para Python (Boto3).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteLoginProfile`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteLoginProfile`.

CLI

AWS CLI

Excluir uma senha de um usuário do IAM

O comando `delete-login-profile` a seguir exclui a senha do usuário do IAM chamado Bob.

```
aws iam delete-login-profile \
    --user-name Bob
```

Este comando não produz saída.

Para obter mais informações, consulte Gerenciamento de senhas de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o perfil de login do usuário do IAM denominado **Bob**. Isso impede que o usuário faça login no console da AWS. Isso não impede que o usuário execute uma CLI, um PowerShell ou chamadas de API da AWS usando chaves de acesso da AWS que ainda possam estar anexadas à conta do usuário.

```
Remove-IAMLoginProfile -UserName Bob
```

- Consulte detalhes da API em DeleteLoginProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o perfil de login do usuário do IAM denominado **Bob**. Isso impede que o usuário faça login no console da AWS. Isso não impede que o usuário execute uma CLI, um PowerShell ou chamadas de API da AWS usando chaves de acesso da AWS que ainda possam estar anexadas à conta do usuário.

```
Remove-IAMLoginProfile -UserName Bob
```

- Para obter detalhes sobre a API, consulte DeleteLognaProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteOpenIdConnectProvider`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteOpenIdConnectProvider`.

CLI

AWS CLI

Excluir um provedor de identidade OpenID Connect do IAM

Este exemplo exclui o provedor OIDC do IAM que se conecta ao provedor `example.oidcprovider.com`.

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Este comando não produz saída.

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o provedor OIDC do IAM que se conecta ao provedor **`example.oidcprovider.com`**. Certifique-se de atualizar ou excluir quaisquer perfis que façam referência a esse provedor no elemento **`Principal`** da política de confiança do perfil.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Consulte detalhes da API em DeleteOpenIdConnectProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o provedor OIDC do IAM que se conecta ao provedor **`example.oidcprovider.com`**. Certifique-se de atualizar ou excluir quaisquer perfis que façam referência a esse provedor no elemento **`Principal`** da política de confiança do perfil.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Para obter detalhes sobre a API, consulte DeleteOpenIdConnectProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeletePolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeletePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Criar usuários somente leitura e leitura/gravação usando
- Como criar um espaço de trabalho de monitoramento gerenciado
- Conceitos básicos de streaming gerenciado
- Conceitos básicos da orquestração de fluxo de trabalaho
- Políticas gerenciadas
- Como configurar o gerenciamento de sistemas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function iecho
#
```

```
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an AWS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
```

```
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}
```

- Consulte detalhes da API em DeletePolicy na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como excluir uma política do IAM

Este exemplo exclui a política cujo ARN é `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeletePolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}
```

```
// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(ctx context.Context, policyArn string)
 error {
 _, err := wrapper.IamClient.DeletePolicy(ctx, &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
```

```
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Exclua a política.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Consulte detalhes da API em DeletePolicy na Referência da API do AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
```

```
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui a política cujo ARN é **`arn:aws:iam::123456789012:policy/MySamplePolicy`**. Antes de excluir a política, exclua primeiro todas as versões, exceto a padrão, executando **`Remove-IAMPolicyVersion`**. Você também deve desassociar a política de qualquer usuário, grupo ou perfil do IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Exemplo 2: este exemplo exclui uma política excluindo primeiro todas as versões não padrão da política, desassociando-a de todas as entidades do IAM anexadas e, por fim, excluindo a própria política. A primeira linha recupera o objeto da política. A segunda linha recupera todas as versões da política que não estão marcadas como a versão padrão em uma compilação e depois exclui cada política na compilação. A terceira linha recupera todos os usuários, grupos e perfis do IAM aos quais a política está anexada. As linhas de quatro a seis desassociam a política de cada entidade anexada. A última linha usa esse comando para remover a política gerenciada e a versão padrão restante. O exemplo inclui o parâmetro switch **`-Force`** em qualquer linha que precise dele para suprimir solicitações de confirmação.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
```

```
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Consulte detalhes da API em DeletePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui a política cujo ARN é **`arn:aws:iam::123456789012:policy/MySamplePolicy`**. Antes de excluir a política, exclua primeiro todas as versões, exceto a padrão, executando **`Remove-IAMPolicyVersion`**. Você também deve desassociar a política de qualquer usuário, grupo ou perfil do IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Exemplo 2: este exemplo exclui uma política excluindo primeiro todas as versões não padrão da política, desassociando-a de todas as entidades do IAM anexadas e, por fim, excluindo a própria política. A primeira linha recupera o objeto da política. A segunda linha recupera todas as versões da política que não estão marcadas como a versão padrão em uma compilação e depois exclui cada política na compilação. A terceira linha recupera todos os usuários, grupos e perfis do IAM aos quais a política está anexada. As linhas de quatro a seis desassociam a política de cada entidade anexada. A última linha usa esse comando para remover a política gerenciada e a versão padrão restante. O exemplo inclui o parâmetro switch **`-Force`** em qualquer linha que precise dele para suprimir solicitações de confirmação.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Para obter detalhes sobre a API, consulte DeletePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK para Python (Boto3).

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Consulte detalhes da API em DeletePolicy na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->deletepolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Policy cannot be deleted due to attachments.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte DeletePolicy na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            print("ERROR: deletePolicy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API do AWS SDK para Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeletePolicyVersion`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeletePolicyVersion`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Políticas gerenciadas
- Reverter uma versão de política

CLI

AWS CLI

Para excluir uma versão de uma política gerenciada

Este exemplo exclui a versão identificada como v2 da política cujo ARN é `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeletePolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui a versão identificada como **v2** da política cujo ARN é **arn:aws:iam::123456789012:policy/MySamplePolicy**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Exemplo 2: este exemplo exclui uma política excluindo primeiro todas as versões não padrão da política e depois excluindo a própria política. A primeira linha recupera o objeto da política. A segunda linha recupera todas as versões da política que não estão marcadas como padrão em uma compilação e, em seguida, usa esse comando para excluir cada política na compilação. A última linha remove a política em si, bem como a versão padrão restante. Observe que, para excluir com êxito uma política gerenciada, você também deve desassociar

a política de qualquer usuário, grupo ou perfis usando os comandos **Unregister-IAMUserPolicy**, **Unregister-IAMGroupPolicy** e **Unregister-IAMRolePolicy**. Veja o exemplo do cmdlet **Remove-IAMPolicy**.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Consulte detalhes da API em DeletePolicyVersion na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui a versão identificada como **v2** da política cujo ARN é **arn:aws:iam::123456789012:policy/MySamplePolicy**.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Exemplo 2: este exemplo exclui uma política excluindo primeiro todas as versões não padrão da política e depois excluindo a própria política. A primeira linha recupera o objeto da política. A segunda linha recupera todas as versões da política que não estão marcadas como padrão em uma compilação e, em seguida, usa esse comando para excluir cada política na compilação. A última linha remove a política em si, bem como a versão padrão restante. Observe que, para excluir com êxito uma política gerenciada, você também deve desassociar a política de qualquer usuário, grupo ou perfis usando os comandos **Unregister-IAMUserPolicy**, **Unregister-IAMGroupPolicy** e **Unregister-IAMRolePolicy**. Veja o exemplo do cmdlet **Remove-IAMPolicy**.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Para obter detalhes sobre a API, consulte DeletePolicyVersion na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    lo_iam->deletepolicyversion(
      iv_policyarn = iv_policy_arn
      iv_versionid = iv_version_id ).
    MESSAGE 'Policy version deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy or version does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamdeleteconflictex.
    MESSAGE 'Cannot delete default policy version.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte DeletePolicyVersion na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteRole`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteRole`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Configurar a conectividade do serviço de contêiner

- Criar uma API REST com integração de proxy de função
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Como criar sua primeira função com tecnologia sem servidor
- Conceitos básicos de data warehouses de tecnologia sem servidor
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos de clusters de data warehouse provisionados
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar funções
- Como mover segredos codificados para o armazenamento seguro de segredos
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
/// <summary>
/// Delete an IAM role.
/// </summary>
/// <param name="roleName">The name of the IAM role to delete.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em DeleteRole na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an AWS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
```

```
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- Consulte detalhes da API em DeleteRole na Referência de comandos da AWS CLI.

CLI

AWS CLI

Como excluir um perfil do IAM

O comando `delete-role`, apresentado a seguir, remove o perfil denominado `Test-Role`.

```
aws iam delete-role \
    --role-name Test-Role
```

Este comando não produz saída.

Antes de poder excluir um perfil, você deve removê-lo de qualquer perfil de instância (`remove-role-from-instance-profile`), desanexar quaisquer políticas gerenciadas

(`detach-role-policy`) e excluir quaisquer políticas em linha anexadas a ele (`delete-role-policy`).

Para obter mais informações, consulte Criação de perfis do IAM e Usar perfis de instância no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(ctx context.Context, roleName string) error
 {
```

```
 _, err := wrapper.IamClient.DeleteRole(ctx, &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Consulte detalhes da API em DeleteRole na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Exclua o perfil.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Consulte detalhes da API em DeleteRole na Referência da API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o perfil denominado **MyNewRole** da conta atual do IAM. Antes de excluir o perfil, use primeiro o comando **Unregister-IAMRolePolicy** para desassociar todas as políticas gerenciadas. As políticas em linha são excluídas com o perfil.

```
Remove-IAMRole -RoleName MyNewRole
```

Exemplo 2: este exemplo desassocia todas as políticas gerenciadas do perfil denominado **MyNewRole** e depois o exclui. A primeira linha recupera todas as políticas gerenciadas anexadas ao perfil como uma compilação e, em seguida, desassocia cada política da compilação do perfil. A segunda linha exclui o perfil em si. As políticas em linha são excluídas com o perfil.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Consulte detalhes da API em DeleteRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o perfil denominado **MyNewRole** da conta atual do IAM. Antes de excluir o perfil, use primeiro o comando **Unregister-IAMRolePolicy** para desassociar todas as políticas gerenciadas. As políticas em linha são excluídas com o perfil.

```
Remove-IAMRole -RoleName MyNewRole
```

Exemplo 2: este exemplo desassocia todas as políticas gerenciadas do perfil denominado **MyNewRole** e depois o exclui. A primeira linha recupera todas as políticas gerenciadas anexadas ao perfil como uma compilação e, em seguida, desassocia cada política da compilação do perfil. A segunda linha exclui o perfil em si. As políticas em linha são excluídas com o perfil.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
```

```
Remove-IAMRole -RoleName MyNewRole
```

- Para obter detalhes sobre a API, consulte DeleteRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Consulte detalhes da API em DeleteRole na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    # Detach and delete attached policies
    @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
      response.attached_policies.each do |policy|
        @iam_client.detach_role_policy({
                                         role_name: role_name,
                                         policy_arn: policy.policy_arn
                                       })
        # Check if the policy is a customer managed policy (not AWS managed)
        unless policy.policy_arn.include?('aws:policy/')
          @iam_client.delete_policy({ policy_arn: policy.policy_arn })
          @logger.info("Deleted customer managed policy #{policy.policy_name}.")
        end
      end
    end

    # Delete the role
    @iam_client.delete_role({ role_name: role_name })
    @logger.info("Deleted role #{role_name}.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't detach policies and delete role #{role_name}. Here's
 why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Consulte detalhes da API em DeleteRole na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Consulte detalhes da API em DeleteRole na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->deleterole( iv_rolename = iv_role_name ).
```

```
        MESSAGE 'Role deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'Role cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte DeleteRole na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            print("ERROR: deleteRole:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API do AWS SDK para Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteRolePermissionsBoundary`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteRolePermissionsBoundary`.

CLI

AWS CLI

Excluir um limite de permissões de um perfil do IAM

O exemplo de `delete-role-permissions-boundary` a seguir exclui o limite de permissões do perfil do IAM especificado. Para aplicar um limite de permissões a um perfil, use o comando `put-role-permissions-boundary`.

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteRolePermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo mostra como remover o limite de permissões anexado a um perfil do IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Consulte detalhes da API em DeleteRolePermissionsBoundary na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo mostra como remover o limite de permissões anexado a um perfil do IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Para obter detalhes sobre a API, consulte DeleteRolePermissionsBoundary na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteRolePolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteRolePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conceitos básicos de data warehouses de tecnologia sem servidor
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de clusters de data warehouse provisionados
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
```

```
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em DeleteRolePolicy na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como remover uma política de um perfil do IAM

O comando `delete-role-policy`, apresentado a seguir, remove a política denominada `ExamplePolicy` do perfil denominado `Test-Role`.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Este comando não produz saída.

Para obter mais informações, consulte Modificar um perfil no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteRolePolicy na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- Consulte detalhes da API em DeleteRolePolicy na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui a política em linha **`S3AccessPolicy`** incorporada no perfil do IAM **`S3BackupRole`**.

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Consulte detalhes da API em DeleteRolePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui a política em linha **`S3AccessPolicy`** incorporada no perfil do IAM **`S3BackupRole`**.

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Para obter detalhes sobre a API, consulte DeleteRolePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteSAMLProvider`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteSAMLProvider`.

CLI

AWS CLI

Como excluir um provedor SAML

Este exemplo exclui o provedor SAML 2.0 do IAM cujo ARN é `arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`.

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Este comando não produz saída.

Para obter mais informações, consulte Criação de provedores de identidade SAML do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteSAMLProvider na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Consulte detalhes da API em DeleteSAMLProvider na Referência da API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o provedor SAML 2.0 do IAM cujo ARN é **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**.

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Consulte detalhes da API em DeleteSAMLProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o provedor SAML 2.0 do IAM cujo ARN é **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**.

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Para obter detalhes sobre a API, consulte DeleteSAMLProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteServerCertificate`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteServerCertificate`.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
```

```
    Aws::IAM::Model::DeleteServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    const auto outcome = iam.DeleteServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error deleting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- Consulte detalhes da API em DeleteServerCertificate na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como excluir um certificado de servidor da sua conta da AWS

O comando `delete-server-certificate`, apresentado a seguir, remove o certificado de servidor especificado da sua conta da AWS.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Este comando não produz saída.

Para listar os certificados de servidor disponíveis em sua conta da AWS, use o comando `list-server-certificates`.

Para obter mais informações, consulte Gerenciar certificados de servidor no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteServerCertificate na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Excluir um certificado de servidor.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.

- Consulte detalhes da API em DeleteServerCertificate na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteServerCertificate na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o certificado do servidor denominado **`MyServerCert`**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Consulte detalhes da API em DeleteServerCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o certificado do servidor denominado **`MyServerCert`**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Para obter detalhes sobre a API, consulte DeleteServerCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Listar, atualizar e excluir certificados de servidor.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
```

```
                                          certificate_body: certificate_body,
                                          private_key: private_key
                                        })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em DeleteServerCertificate na Referência da API AWS SDK para Ruby.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteServiceLinkedRole`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteServiceLinkedRole`.

CLI

AWS CLI

Como excluir um perfil vinculado ao serviço

O exemplo para `delete-service-linked-role`, apresentado a seguir, exclui o perfil vinculado ao serviço especificado que não é mais necessário. A exclusão acontece de forma assíncrona. É possível verificar o status da exclusão e confirmar quando ela for concluída ao usar o comando `get-service-linked-role-deletion-status`.

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Resultado:

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Para obter mais informações, consulte Usar perfis vinculados ao serviço no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteServiceLinkedRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(ctx context.Context, roleName
 string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(ctx,
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
```

```
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- Para obter detalhes da API, consulte DeleteServiceLinkedRole na Referência da API AWS SDK para Go SDK for Kotlin.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Consulte detalhes da API em DeleteServiceLinkedRole na Referência de API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo excluiu o perfil vinculado ao serviço. Observe que, se o serviço ainda estiver usando esse perfil, esse comando resultará em uma falha.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Consulte detalhes da API em DeleteServiceLinkedRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo excluiu o perfil vinculado ao serviço. Observe que, se o serviço ainda estiver usando esse perfil, esse comando resultará em uma falha.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Para obter detalhes sobre a API, consulte DeleteServiceLnakedRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# Deletes a service-linked role.
#
# @param role_name [String] The name of the role to delete.
def delete_service_linked_role(role_name)
  response = @iam_client.delete_service_linked_role(role_name: role_name)
```

```
    task_id = response.deletion_task_id
    check_deletion_status(role_name, task_id)
  rescue Aws::Errors::ServiceError => e
    handle_deletion_error(e, role_name)
  end

  private

  # Checks the deletion status of a service-linked role
  #
  # @param role_name [String] The name of the role being deleted
  # @param task_id [String] The task ID for the deletion process
  def check_deletion_status(role_name, task_id)
    loop do
      response = @iam_client.get_service_linked_role_deletion_status(
        deletion_task_id: task_id
      )
      status = response.status
      @logger.info("Deletion of #{role_name} #{status}.")
      break if %w[SUCCEEDED FAILED].include?(status)

      sleep(3)
    end
  end

  # Handles deletion error
  #
  # @param e [Aws::Errors::ServiceError] The error encountered during deletion
  # @param role_name [String] The name of the role attempted to delete
  def handle_deletion_error(e, role_name)
    return if e.code == 'NoSuchEntity'

    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Para obter detalhes da API, consulte DeleteServiceLinkedRole na Referência da API AWS SDK para Ruby SDK for Kotlin.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Consulte detalhes da API em DeleteServiceLinkedRole na Referência da API AWS SDK for Rust.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteSigningCertificate`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteSigningCertificate`.

CLI

AWS CLI

Excluir um certificado de assinatura de um usuário do IAM

O comando `delete-signing-certificate` a seguir exclui o certificado de assinatura especificado do usuário do IAM chamado Bob.

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

Este comando não produz saída.

Para obter o ID de um certificado de assinatura, use o comando `list-signing-certificates`.

Para obter mais informações, consulte Manage signing certificates no Guia do usuário do Amazon EC2.

- Consulte detalhes da API em DeleteSigningCertificate na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o certificado de assinatura com o ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** do usuário do IAM chamado **Bob**.

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Consulte detalhes da API em DeleteSigningCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o certificado de assinatura com o ID **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** do usuário do IAM chamado **Bob**.

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Para obter detalhes sobre a API, consulte DeleteSignnagCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteUser`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Criar usuários somente leitura e leitura/gravação usando

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an AWS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
```

```
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Consulte detalhes da API em DeleteUser na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
```

```
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como excluir um usuário do IAM

O comando `delete-user`, apresentado a seguir, remove o usuário do IAM denominado Bob da conta atual.

```
aws iam delete-user \
    --user-name Bob
```

Este comando não produz saída.

Para obter mais informações, consulte Exclusão de um usuário do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteUser na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
```

```
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(ctx context.Context, userName string) error
 {
 _, err := wrapper.IamClient.DeleteUser(ctx, &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
```

```
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Exclua o usuário.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun deleteIAMUser(userNameVal: String) {
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o usuário do IAM chamado **Bob**.

```
Remove-IAMUser -UserName Bob
```

Exemplo 2: este exemplo exclui a usuária do IAM chamada **Theresa** com todos os elementos que devem ser excluídos primeiro.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }

# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

```
# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Consulte detalhes da API em DeleteUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o usuário do IAM chamado **Bob**.

```
Remove-IAMUser -UserName Bob
```

Exemplo 2: este exemplo exclui a usuária do IAM chamada **Theresa** com todos os elementos que devem ser excluídos primeiro.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }
```

```
# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Para obter detalhes sobre a API, consulte DeleteUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
```

```
        raise
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- Consulte detalhes da API em DeleteUser na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->deleteuser( iv_username = iv_user_name ).
        MESSAGE 'User deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamdeleteconflictex.
        MESSAGE 'User cannot be deleted due to attached resources.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte DeleteUser na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func deleteUser(user: IAMClientTypes.User) async throws {
        let input = DeleteUserInput(
```

```
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUser(input: input)
        } catch {
            print("ERROR: deleteUser:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API do AWS SDK para Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteUserPermissionsBoundary`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteUserPermissionsBoundary`.

CLI

AWS CLI

Excluir um limite de permissões de um usuário do IAM

O exemplo `delete-user-permissions-boundary` a seguir exclui o limite de permissões anexado ao usuário do IAM chamado `intern`. Para aplicar um limite de permissões a um usuário, use o comando `put-user-permissions-boundary`.

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

Este comando não produz saída.

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteUserPermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo mostra como remover o limite de permissões anexado a um usuário do IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Consulte detalhes da API em DeleteUserPermissionsBoundary na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo mostra como remover o limite de permissões anexado a um usuário do IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Para obter detalhes sobre a API, consulte DeleteUserPermissionsBoundary na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteUserPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteUserPolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conheça os conceitos básicos

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em DeleteUserPolicy na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como remover uma política de um usuário do IAM

O comando `delete-user-policy`, apresentado a seguir, remove a política especificada do usuário do IAM denominado Bob.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Este comando não produz saída.

Para obter uma lista de políticas para um usuário do IAM, use o comando `list-user-policies`.

Para obter mais informações, consulte Criar um usuário do IAM na sua conta da AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteUserPolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
```

```
func (wrapper UserWrapper) DeleteUserPolicy(ctx context.Context, userName string,
 policyName string) error {
 _, err := wrapper.IamClient.DeleteUserPolicy(ctx, &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Consulte detalhes da API em DeleteUserPolicy na Referência da API AWS SDK para Go.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui a política em linha denominada **`AccessToEC2Policy`** incorporada no usuário do IAM chamado **`Bob`**.

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Exemplo 2: este exemplo encontra todas as políticas em linha incorporadas na usuária do IAM chamada **`Theresa`** e depois as exclui.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Consulte detalhes da API em DeleteUserPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui a política em linha denominada **`AccessToEC2Policy`** incorporada no usuário do IAM chamado **`Bob`**.

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Exemplo 2: este exemplo encontra todas as políticas em linha incorporadas na usuária do IAM chamada **`Theresa`** e depois as exclui.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Para obter detalhes sobre a API, consulte DeleteUserPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Consulte detalhes da API em DeleteUserPolicy na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- Consulte detalhes da API em DeleteUserPolicy na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
 throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            print("ERROR: deleteUserPolicy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte DeleteUserPolicy na Referência da API do AWS SDK para Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteVirtualMfaDevice`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DeleteVirtualMfaDevice`.

CLI

AWS CLI

Remover um dispositivo de MFA virtual

O comando `delete-virtual-mfa-device` a seguir remove o dispositivo de MFA especificado da conta atual.

```
aws iam delete-virtual-mfa-device \
```

```
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

Este comando não produz saída.

Para obter mais informações, consulte Desativar dispositivos de MFA no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DeleteVirtualMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o dispositivo de MFA virtual do IAM cujo ARN é **arn:aws:iam::123456789012:mfa/bob**.

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Exemplo 2: este exemplo verifica se a usuária do IAM Theresa tem um dispositivo de MFA atribuído. Se for encontrado, o dispositivo é desabilitado para a usuária do IAM. Se o dispositivo for virtual, ele também é excluído.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Consulte detalhes da API em DeleteVirtualMfaDevice na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o dispositivo de MFA virtual do IAM cujo ARN é **arn:aws:iam::123456789012:mfa/bob**.

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Exemplo 2: este exemplo verifica se a usuária do IAM Theresa tem um dispositivo de MFA atribuído. Se for encontrado, o dispositivo é desabilitado para a usuária do IAM. Se o dispositivo for virtual, ele também é excluído.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Para obter detalhes sobre a API, consulte DeleteVirtualMfaDevice na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DetachGroupPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DetachGroupPolicy`.

CLI

AWS CLI

Para desanexar uma política de um grupo

Este exemplo remove a política gerenciada com o ARN `arn:aws:iam::123456789012:policy/TesterAccessPolicy` do grupo denominado `Testers`.

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Este comando não produz saída.

Para obter mais informações, consulte Gerenciar grupos de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DetachGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo desassocia a política de grupo gerenciado cujo ARN é **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** do grupo denominado **`Testers`**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Exemplo 2: este exemplo encontra todas as políticas gerenciadas que estão anexadas ao grupo denominado **`Testers`** e as desassocia do grupo.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Consulte detalhes da API em DetachGroupPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo desassocia a política de grupo gerenciado cujo ARN é **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** do grupo denominado **`Testers`**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Exemplo 2: este exemplo encontra todas as políticas gerenciadas que estão anexadas ao grupo denominado **`Testers`** e as desassocia do grupo.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Para obter detalhes sobre a API, consulte DetachGroupPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DetachRolePolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DetachRolePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conheça os conceitos básicos
- Configurar a conectividade do serviço de contêiner
- Criar uma API REST com integração de proxy de função
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Como criar sua primeira função com tecnologia sem servidor
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar funções
- Como mover segredos codificados para o armazenamento seguro de segredos
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Para obter detalhes da API, consulte.DetachRolePolicy na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
 "
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como desanexar uma política de um perfil

Este exemplo remove a política gerenciada com o ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` do perfil denominado `FedTesterRole`.

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Este comando não produz saída.

Para obter mais informações, consulte Modificar um perfil no Guia do usuário do AWS IAM.

- Para obter detalhes da API, consulte.DetachRolePolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(ctx context.Context, roleName string,
 policyArn string) error {
 _, err := wrapper.IamClient.DetachRolePolicy(ctx, &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }
```

```
    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Desanexe a política.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
```

```
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String,
) {
```

```
    val request =
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- Consulte detalhes da API em DetachRolePolicy na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo desassocia a política de grupo gerenciado cujo ARN é **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** do perfil denominado **`FedTesterRole`**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Exemplo 2: este exemplo encontra todas as políticas gerenciadas que estão anexadas ao perfil denominado **`FedTesterRole`** e as desassocia dele.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Consulte detalhes da API em DetachRolePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo desassocia a política de grupo gerenciado cujo ARN é **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** do perfil denominado **`FedTesterRole`**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Exemplo 2: este exemplo encontra todas as políticas gerenciadas que estão anexadas ao perfil denominado **`FedTesterRole`** e as desassocia dele.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Para obter detalhes sobre a API, consulte DetachRolePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Desanexar uma política de uma função usando o objeto Policy do Boto3.

```
def detach_from_role(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Desanexar uma política de uma função usando o objeto Role do Boto3.

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Consulte detalhes da API em DetachRolePolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, anexa e desconecta políticas de perfis.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
 #{e.message}")
```

```
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
```

```
    false
  end
end
```

- Para obter detalhes da API, consulte.DetachRolePolicy na AWS SDK para RubyReferência da API do.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Consulte detalhes da API em DetachRolePolicy na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->detachrolepolicy(
          iv_rolename = iv_role_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy detached from role successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role or policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte DetachRolePolicy na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
 IAMClientTypes.Role) async throws {
        let input = DetachRolePolicyInput(
```

```
            policyArn: policy.arn,
            roleName: role.roleName
        )

        do {
            _ = try await iamClient.detachRolePolicy(input: input)
        } catch {
            print("ERROR: detachRolePolicy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte DetachRolePolicy na Referência da API do AWS SDK for Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DetachUserPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DetachUserPolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Criar usuários somente leitura e leitura/gravação usando

CLI

AWS CLI

Como desanexar uma política de um usuário

Este exemplo remove a política gerenciada com o ARN `arn:aws:iam::123456789012:policy/TesterPolicy` do usuário Bob.

```
aws iam detach-user-policy \
    --user-name Bob \
```

```
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Este comando não produz saída.

Para obter mais informações, consulte Alteração de permissões de um usuário do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DetachUserPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo desassocia a política gerenciada cujo ARN é **arn:aws:iam::123456789012:policy/TesterPolicy** do usuário do IAM chamado **Bob**.

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemplo 2: este exemplo encontra todas as políticas gerenciadas que estão anexadas ao usuário do IAM chamado **Theresa** e as desassocia dele.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Consulte detalhes da API em DetachUserPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo desassocia a política gerenciada cujo ARN é **arn:aws:iam::123456789012:policy/TesterPolicy** do usuário do IAM chamado **Bob**.

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemplo 2: este exemplo encontra todas as políticas gerenciadas que estão anexadas ao usuário do IAM chamado **Theresa** e as desassocia dele.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Para obter detalhes sobre a API, consulte DetachUserPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- Consulte detalhes da API em DetachUserPolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
   )
   @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
   true
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error('Error detaching policy: Policy or user does not exist.')
   false
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error detaching policy from user '#{user_name}':
#{e.message}")
   false
 end
```

- Para obter mais detalhes da API, consulte DetachUserPolicy na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Consulte detalhes da API em DetachUserPolicy na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->detachuserpolicy(
```

```
          iv_username = iv_user_name
          iv_policyarn = iv_policy_arn ).
        MESSAGE 'Policy detached from user successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User or policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte DetachUserPolicy na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`EnableMfaDevice`** com uma CLI

Os exemplos de código a seguir mostram como usar o `EnableMfaDevice`.

CLI

AWS CLI

Habilitar um dispositivo de MFA

Depois de usar o comando `create-virtual-mfa-device` para criar um dispositivo de MFA virtual, você pode atribuir o dispositivo de MFA a um usuário. O exemplo de `enable-mfa-device` a seguir atribui o dispositivo de MFA com o número de série `arn:aws:iam::210987654321:mfa/BobsMFADevice` ao usuário Bob. O comando também sincroniza o dispositivo com a AWS incluindo os dois primeiros códigos em sequência do dispositivo de MFA virtual.

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

Este comando não produz saída.

Para obter mais informações, consulte Habilitar um dispositivo de autenticação multifator (MFA) virtual no Guia do usuário do AWS IAM.

- Consulte detalhes da API em EnableMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando habilita o dispositivo de MFA de hardware com o número de série **987654321098** e associa o dispositivo ao usuário **Bob**. Ele inclui os dois primeiros códigos em sequência do dispositivo.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Exemplo 2: este exemplo cria e habilita um dispositivo de MFA virtual. O primeiro comando cria o dispositivo virtual e retorna a representação de objeto do dispositivo na variável **$MFADevice**. Você pode usar as propriedades **.Base32StringSeed** ou **QRCodePng** para configurar a aplicação de software do usuário. O comando final atribui o dispositivo ao usuário **David**, identificando o dispositivo pelo número de série. O comando também sincroniza o dispositivo com a AWS incluindo os dois primeiros códigos em sequência do dispositivo de MFA virtual.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Consulte detalhes da API em EnableMfaDevice na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando habilita o dispositivo de MFA de hardware com o número de série **987654321098** e associa o dispositivo ao usuário **Bob**. Ele inclui os dois primeiros códigos em sequência do dispositivo.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Exemplo 2: este exemplo cria e habilita um dispositivo de MFA virtual. O primeiro comando cria o dispositivo virtual e retorna a representação de objeto do dispositivo na variável **`$MFADevice`**. Você pode usar as propriedades **`.Base32StringSeed`** ou **`QRCodePng`** para configurar a aplicação de software do usuário. O comando final atribui o dispositivo ao usuário **`David`**, identificando o dispositivo pelo número de série. O comando também sincroniza o dispositivo com a AWS incluindo os dois primeiros códigos em sequência do dispositivo de MFA virtual.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Para obter detalhes sobre a API, consulte EnableMfaDevice na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GenerateCredentialReport`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GenerateCredentialReport`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

CLI

AWS CLI

Como gerar um relatório de credenciais

O exemplo apresentado a seguir tenta gerar um relatório de credenciais para a conta da AWS.

```
aws iam generate-credential-report
```

Resultado:

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Para obter mais informações, consulte Obter relatórios de credenciais da sua conta da AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GenerateCredentialReport na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo solicita a geração de um novo relatório, que pode ser feito a cada quatro horas. Se o último relatório ainda for recente, o campo Estado será **COMPLETE**. Use **Get-IAMCredentialReport** para visualizar o relatório completo.

```
Request-IAMCredentialReport
```

Saída:

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- Consulte detalhes da API em GenerateCredentialReport na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo solicita a geração de um novo relatório, que pode ser feito a cada quatro horas. Se o último relatório ainda for recente, o campo Estado será **COMPLETE**. Use **Get-IAMCredentialReport** para visualizar o relatório completo.

```
Request-IAMCredentialReport
```

Saída:

```
Description                                                    State
-----------                                                    -----
No report exists. Starting a new report generation task        STARTED
```

- Para obter detalhes sobre a API, consulte GenerateCredentialReport na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
```

```
    else:
        return response
```

- Consulte detalhes da API em GenerateCredentialReport na Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->generatecredentialreport( ).
    MESSAGE 'Credential report generation started.' TYPE 'I'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Report generation limit exceeded.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when generating credential report.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte GenerateCredentialReport na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GenerateServiceLastAccessedDetails`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GenerateServiceLastAccessedDetails`.

CLI

AWS CLI

Exemplo 1: para gerar um relatório de acesso ao serviço de uma política personalizada

O exemplo de `generate-service-last-accessed-details` a seguir inicia um trabalho em segundo plano para gerar um relatório que lista os serviços acessados pelos usuários do IAM e outras entidades com uma política personalizada denominada `intern-boundary`. Você pode exibir o relatório após a criação executando o comando `get-service-last-accessed-details`.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Resultado:

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

Exemplo 2: para gerar um relatório de acesso ao serviço da política gerenciada AdministratorAccess da AWS

O exemplo de `generate-service-last-accessed-details` a seguir inicia um trabalho em segundo plano para gerar um relatório que lista os serviços acessados pelos usuários do IAM e outras entidades com a política gerenciada `AdministratorAccess` da AWS. Você pode exibir o relatório após a criação executando o comando `get-service-last-accessed-details`.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Resultado:

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

Para obter mais informações, consulte Refinar permissões na AWS usando as informações do último acesso no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GenerateServiceLastAccessedDetails na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo é um cmdlet equivalente da API GenerateServiceLastAccessedDetails. Isso fornece um ID de trabalho que pode ser usado em Get-IAMServiceLastAccessedDetail and Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Consulte detalhes da API em GenerateServiceLastAccessedDetails na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo é um cmdlet equivalente da API GenerateServiceLastAccessedDetails. Isso fornece um ID de trabalho que pode ser usado em Get-IAMServiceLastAccessedDetail and Get-IAMServiceLastAccessedDetailWithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Para obter detalhes sobre a API, consulte GenerateServiceLastAccessedDetails na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetAccessKeyLastUsed`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetAccessKeyLastUsed`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar chaves de acesso

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);

    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em GetAccessKeyLastUsed na Referência da API AWS SDK para C++.

CLI

AWS CLI

Para recuperar informações sobre quando a chave de acesso especificada foi usada pela última vez

O exemplo apresentado a seguir recupera informações sobre quando a chave de acesso ABCDEXAMPLE foi usada pela última vez.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Resultado:

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Para obter mais informações, consulte Gerenciamento de chaves de acesso de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetAccessKeyLastUsed na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha a chave de acesso.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.

- Consulte detalhes da API em GetAccessKeyLastUsed na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em GetAccessKeyLastUsed na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: retorna o nome de usuário proprietário e as informações do último uso da chave de acesso fornecida.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Para ver detalhes da API, consulte GetAccessKeyLastUsed na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: retorna o nome de usuário proprietário e as informações do último uso da chave de acesso fornecida.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Para obter detalhes sobre a API, consulte GetAccessKeyLastUsed na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
```

```
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- Consulte detalhes da API em GetAccessKeyLastUsed na Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->getaccesskeylastused(
          iv_accesskeyid = iv_access_key_id ).
        MESSAGE 'Retrieved access key last used information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Access key does not exist.' TYPE 'E'.
    ENDTRY.
```

- Consulte detalhes da API em GetAccessKeyLastUsed na Referência da API AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetAccountAuthorizationDetails`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetAccountAuthorizationDetails`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

CLI

AWS CLI

Para listar usuários, grupos, perfis e políticas do IAM de contas da AWS

O comando `get-account-authorization-details`, apresentado a seguir, retorna informações sobre todos os usuários, grupos, perfis e políticas do IAM na conta da AWS.

```
aws iam get-account-authorization-details
```

Resultado:

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
```

```
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2014-07-30T17:09:20Z",
        "InstanceProfileList": [
            {
                "InstanceProfileId": "AIPA1234567890EXAMPLE",
                "Roles": [
                    {
                        "AssumeRolePolicyDocument": {
                            "Version":"2012-10-17",
                            "Statement": [
                                {
                                    "Sid": "",
                                    "Effect": "Allow",
                                    "Principal": {
                                        "Service": "ec2.amazonaws.com"
                                    },
                                    "Action": "sts:AssumeRole"
                                }
                            ]
                        },
                        "RoleId": "AROA1234567890EXAMPLE",
                        "CreateDate": "2014-07-30T17:09:20Z",
                        "RoleName": "EC2role",
                        "Path": "/",
                        "Arn": "arn:aws:iam::123456789012:role/EC2role"
                    }
                ],
                "CreateDate": "2014-07-30T17:09:20Z",
                "InstanceProfileName": "EC2role",
                "Path": "/",
                "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
            }
        ],
        "RoleName": "EC2role",
        "Path": "/",
        "AttachedManagedPolicies": [
            {
                "PolicyName": "AmazonS3FullAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
            },
```

```
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
```

```
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
```

```
                    "Version":"2012-10-17",
                    "Statement": {
                        "Effect": "Deny",
                        "Action": [
                            "aws-portal:*",
                            "iam:*"
                        ],
                        "Resource": "*"
                    }
                }
            }
        ],
        "Path": "/",
        "AttachedManagedPolicies": [],
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    },
    {
        "UserName": "Charlie",
        "GroupList": [
            "Dev"
        ],
        "CreateDate": "2013-10-14T18:33:56Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [],
        "Path": "/",
        "AttachedManagedPolicies": [],
        "Arn": "arn:aws:iam::123456789012:user/Charlie"
    }
],
"Policies": [
    {
        "PolicyName": "create-update-delete-set-managed-policies",
        "CreateDate": "2015-02-06T19:58:34Z",
        "AttachmentCount": 1,
        "IsAttachable": true,
        "PolicyId": "ANPA1234567890EXAMPLE",
        "DefaultVersionId": "v1",
        "PolicyVersionList": [
            {
                "CreateDate": "2015-02-06T19:58:34Z",
                "VersionId": "v1",
                "Document": {
                    "Version":"2012-10-17",
                    "Statement": {
```

```
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
```

```
                                "arn:aws:s3:::amzn-s3-demo-bucket",
                                "arn:aws:s3:::amzn-s3-demo-bucket/*"
                            ]
                        }
                    ]
                },
                "IsDefaultVersion": true
            }
        ],
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
        "UpdateDate": "2015-01-21T23:39:41Z"
    },
    {
        "PolicyName": "AmazonEC2FullAccess",
        "CreateDate": "2015-02-06T18:40:15Z",
        "AttachmentCount": 1,
        "IsAttachable": true,
        "PolicyId": "ANPA1234567890EXAMPLE",
        "DefaultVersionId": "v1",
        "PolicyVersionList": [
            {
                "CreateDate": "2014-10-30T20:59:46Z",
                "VersionId": "v1",
                "Document": {
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Action": "ec2:*",
                            "Effect": "Allow",
                            "Resource": "*"
                        },
                        {
                            "Effect": "Allow",
                            "Action": "elasticloadbalancing:*",
                            "Resource": "*"
                        },
                        {
                            "Effect": "Allow",
                            "Action": "cloudwatch:*",
                            "Resource": "*"
                        },
                        {
```

```
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Para obter mais informações, consulte Diretrizes de auditoria de segurança da AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetAccountAuthorizationDetails na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo obtém detalhes de autorização sobre as identidades na conta da AWS e exibe a lista de elementos do objeto retornado, incluindo usuários, grupos e perfis. Por exemplo, a propriedade **`UserDetailList`** exibe detalhes sobre os usuários. Informações semelhantes estão disponíveis nas propriedades **`RoleDetailList`** e **`GroupDetailList`**.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Saída:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
```

```
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Saída:

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Consulte detalhes da API em GetAccountAuthorizationDetails na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo obtém detalhes de autorização sobre as identidades na conta da AWS e exibe a lista de elementos do objeto retornado, incluindo usuários, grupos e perfis. Por exemplo, a propriedade **`UserDetailList`** exibe detalhes sobre os usuários. Informações semelhantes estão disponíveis nas propriedades **`RoleDetailList`** e **`GroupDetailList`**.

```
$Details=Get-IAMAccountAuthorizationDetail
```

```
$Details
```

Saída:

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Saída:

```
Arn             : arn:aws:iam::123456789012:user/Administrator
CreateDate      : 10/16/2014 9:03:09 AM
GroupList       : {Administrators}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE1
UserName        : Administrator
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/Bob
CreateDate      : 4/6/2015 12:54:42 PM
GroupList       : {Developers}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE2
UserName        : bab
UserPolicyList : {}

Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Para obter detalhes sobre a API, consulte GetAccountAuthorizationDetails na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Consulte detalhes da API em GetAccountAuthorizationDetails na Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
   TRY.
       oo_result = lo_iam->getaccountauthdetails( ).
       MESSAGE 'Retrieved account authorization details.' TYPE 'I'.
     CATCH /aws1/cx_iamservicefailureex.
       MESSAGE 'Service failure when getting account authorization details.'
TYPE 'E'.
   ENDTRY.
```

- Para obter detalhes da API, consulte GetAccountAuthorizationDetails na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetAccountPasswordPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetAccountPasswordPolicy`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como visualizar a política de senha da conta atual

O comando `get-account-password-policy`, apresentado a seguir, exibe detalhes sobre a política de senha para a conta atual.

```
aws iam get-account-password-policy
```

Resultado:

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Se nenhuma política de senha estiver definida para a conta, o comando retornará um erro `NoSuchEntity`.

Para obter mais informações, consulte Definição de uma política de senhas de contas para usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
```

```
func (wrapper AccountWrapper) GetAccountPasswordPolicy(ctx context.Context)
 (*types.PasswordPolicy, error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(ctx,
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha a política de senha da conta.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
```

```
  console.log(response.PasswordPolicy);
  return response;
};
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes sobre a política de senha da conta atual. Se nenhuma política de senha estiver definida na conta, o comando retorna um erro **`NoSuchEntity`**.

```
Get-IAMAccountPasswordPolicy
```

Saída:

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Para ver detalhes da API, consulte GetAccountPasswordPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes sobre a política de senha da conta atual. Se nenhuma política de senha estiver definida na conta, o comando retorna um erro **`NoSuchEntity`**.

```
Get-IAMAccountPasswordPolicy
```

Saída:

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Para obter detalhes sobre a API, consulte GetAccountPasswordPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
 {pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
 {pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
 {pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
 {pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
```

```
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'IAMPolicyManager'
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    response = @iam_client.get_account_password_policy
    @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.info('The account does not have a password policy.')
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't print the account password policy. Error: #{e.code} -
 #{e.message}")
```

```
    raise
  end
end
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Consulte detalhes da API em GetAccountPasswordPolicy na Referência da API AWS SDK para Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->getaccountpasswordpolicy( ).
        MESSAGE 'Retrieved account password policy.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'No password policy exists.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting password policy.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetAccountSummary`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetAccountSummary`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

CLI

AWS CLI

Como obter informações sobre o uso da entidade do IAM e das cotas do IAM na conta atual

O comando `get-account-summary`, apresentado a seguir, retorna informações sobre o uso atual da entidade do IAM e das cotas atuais da entidade do IAM na conta.

```
aws iam get-account-summary
```

Resultado:

```
{
    "SummaryMap": {
```

```
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Para obter mais informações sobre as limitações de entidade, consulte IAM e cotas do AWS STS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetAccountSummary na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna informações sobre o uso atual da entidade do IAM e das cotas atuais da entidade do IAM na Conta da AWS.

```
Get-IAMAccountSummary
```

Saída:

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
UsersQuota                              5000
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- Para ver detalhes da API, consulte GetAccountSummary na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna informações sobre o uso atual da entidade do IAM e das cotas atuais da entidade do IAM na Conta da AWS.

```
Get-IAMAccountSummary
```

Saída:

```
Key                                     Value
Users                                   7
GroupPolicySizeQuota                    5120
PolicyVersionsInUseQuota                10000
ServerCertificatesQuota                 20
AccountSigningCertificatesPresent       0
AccountAccessKeysPresent                0
Groups                                  3
UsersQuota                              5000
RolePolicySizeQuota                     10240
UserPolicySizeQuota                     2048
GroupsPerUserQuota                      10
AssumeRolePolicySizeQuota               2048
AttachedPoliciesPerGroupQuota           2
Roles                                   9
VersionsPerPolicyQuota                  5
GroupsQuota                             100
PolicySizeQuota                         5120
Policies                                5
RolesQuota                              250
ServerCertificates                      0
AttachedPoliciesPerRoleQuota            2
MFADevicesInUse                         2
PoliciesQuota                           1000
AccountMFAEnabled                       1
Providers                               2
InstanceProfilesQuota                   100
MFADevices                              4
AccessKeysPerUserQuota                  2
AttachedPoliciesPerUserQuota            2
SigningCertificatesPerUserQuota         2
PolicyVersionsInUse                     4
InstanceProfiles                        1
...
```

- Para obter detalhes sobre a API, consulte GetAccountSummary na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Consulte detalhes da API em GetAccountSummary na Referência da API AWS SDK para Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->getaccountsummary( ).
        MESSAGE 'Retrieved account summary.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when getting account summary.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte GetAccountSummary na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetContextKeysForCustomPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetContextKeysForCustomPolicy`.

CLI

AWS CLI

Exemplo 1: para listar as chaves de contexto referenciadas por uma ou mais políticas JSON personalizadas fornecidas como um parâmetro na linha de comando

O comando `get-context-keys-for-custom-policy` a seguir analisa cada política fornecida e lista as chaves de contexto usadas por essas políticas. Use esse comando para identificar quais valores de chave de contexto você deve fornecer para usar com êxito os comandos do simulador de políticas `simulate-custom-policy` e `simulate-custom-policy`. Você também pode recuperar a lista de chaves de contexto utilizadas por todas as políticas associadas a um perfil ou usuário do IAM com o comando `get-context-keys-for-custom-policy`. Os valores de parâmetro que começam com `file://` instruem o comando a ler o arquivo e usar o conteúdo como o valor do parâmetro em vez do próprio nome do arquivo.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
```

```
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}'
```

Resultado:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Exemplo 2: para listar as chaves de contexto referenciadas por uma ou mais políticas JSON personalizadas fornecidas como entrada de arquivo

O comando `get-context-keys-for-custom-policy` a seguir é igual ao exemplo anterior, exceto que as políticas são fornecidas em um arquivo e não como um parâmetro. Como o comando espera uma lista JSON de strings e não uma lista de estruturas JSON, o arquivo deve ser estruturado da forma a seguir, embora você possa reduzi-lo em uma só.

```
[
    "Policy1",
    "Policy2"
]
```

Assim, por exemplo, um arquivo que contém a política do exemplo anterior deve ter a aparência a seguir. Você deve escapar cada aspas duplas incorporadas dentro da string de política precedendo-as com uma barra invertida ".

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

Esse arquivo pode então ser enviado ao comando a seguir.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

Resultado:

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Para obter mais informações, consulte Uso do simulador de políticas do IAM (AWS CLI e API da AWS) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetContextKeysForCustomPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo busca todas as chaves de contexto presentes no JSON da política fornecida. A fim de produzir várias políticas, você pode fornecer uma lista de valores separados por vírgula.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Consulte detalhes da API em GetContextKeysForCustomPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo busca todas as chaves de contexto presentes no JSON da política fornecida. A fim de produzir várias políticas, você pode fornecer uma lista de valores separados por vírgula.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
```

```
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Para obter detalhes sobre a API, consulte GetContextKeysForCustomPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetContextKeysForPrincipalPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetContextKeysForPrincipalPolicy`.

CLI

AWS CLI

Para listar as chaves de contexto referenciadas por todas as políticas associadas a uma entidade principal do IAM

O comando `get-context-keys-for-principal-policy` a seguir recupera todas as políticas anexadas à usuária `saanvi` e aos grupos dos quais ela é membro. Em seguida, ele analisa cada uma delas e lista as chaves de contexto usadas por essas políticas. Utilize esse comando para identificar quais valores de chave de contexto você deve fornecer para usar com êxito os comandos `simulate-custom-policy` e `simulate-principal-policy`. Você também pode recuperar a lista de chaves de contexto usadas por uma política JSON arbitrária com o comando `get-context-keys-for-custom-policy`.

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

Resultado:

```
{
```

```
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Para obter mais informações, consulte Uso do simulador de políticas do IAM (AWS CLI e API da AWS) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetContextKeysForPrincipalPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo busca todas as chaves de contexto presentes no JSON da política fornecida e as políticas anexadas à entidade do IAM (usuário, perfil etc.). Em -PolicyInputList, você pode fornecer uma lista de múltiplos valores como valores separados por vírgula.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Consulte detalhes da API em GetContextKeysForPrincipalPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo busca todas as chaves de contexto presentes no JSON da política fornecida e as políticas anexadas à entidade do IAM (usuário, perfil etc.). Em -PolicyInputList, você pode fornecer uma lista de múltiplos valores como valores separados por vírgula.

```
$policy1 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
```

```
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17",       "Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Para obter detalhes sobre a API, consulte GetContextKeysForPrincipalPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetCredentialReport`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetCredentialReport`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

CLI

AWS CLI

Como obter um relatório de credenciais

Este exemplo abre o relatório retornado e o envia ao pipeline como uma matriz de linhas de texto.

```
aws iam get-credential-report
```

Resultado:

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
```

```
}
```

Para obter mais informações, consulte Obter relatórios de credenciais da sua conta da AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetCredentialReport na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo1: este exemplo abre o relatório retornado e o envia ao pipeline como uma matriz de linhas de texto. A primeira linha é o cabeçalho com nomes de colunas separados por vírgula. Cada linha sucessiva é a linha de detalhes de um usuário, com cada campo separado por vírgulas. Antes de visualizar o relatório, você deve gerá-lo com o cmdlet **`Request-IAMCredentialReport`**. Para recuperar o relatório como uma única string, use **`-Raw`** em vez de **`-AsTextArray`**. O alias **`-SplitLines`** também é aceito no switch **`-AsTextArray`**. Para obter a lista completa de colunas na saída, consulte a referência de API do serviço. Observe que, se não usar **`-AsTextArray`** ou **`-SplitLines`**, você deve extrair o texto da propriedade **`.Content`** usando a classe **`StreamReader`** .NET.

```
Request-IAMCredentialReport
```

Saída:

```
Description                                                          State
-----------                                                          -----
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Saída:

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
```

```
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Para ver detalhes da API, consulte GetCredentialReport na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo1: este exemplo abre o relatório retornado e o envia ao pipeline como uma matriz de linhas de texto. A primeira linha é o cabeçalho com nomes de colunas separados por vírgula. Cada linha sucessiva é a linha de detalhes de um usuário, com cada campo separado por vírgulas. Antes de visualizar o relatório, você deve gerá-lo com o cmdlet **`Request-IAMCredentialReport`**. Para recuperar o relatório como uma única string, use **`-Raw`** em vez de **`-AsTextArray`**. O alias **`-SplitLines`** também é aceito no switch **`-AsTextArray`**. Para obter a lista completa de colunas na saída, consulte a referência de API do serviço. Observe que, se não usar **`-AsTextArray`** ou **`-SplitLines`**, você deve extrair o texto da propriedade **`.Content`** usando a classe **`StreamReader`** .NET.

```
Request-IAMCredentialReport
```

Saída:

```
Description                                                          State
-----------                                                          -----
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Saída:

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
```

```
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Para obter detalhes sobre a API, consulte GetCredentialReport na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Consulte detalhes da API em GetCredentialReport na Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->getcredentialreport( ).
    MESSAGE 'Retrieved credential report.' TYPE 'I'.
  CATCH /aws1/cx_iamcredrptnotpresen00.
    MESSAGE 'Credential report not present.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptexpiredex.
    MESSAGE 'Credential report expired.' TYPE 'E'.
  CATCH /aws1/cx_iamcredrptnotreadyex.
    MESSAGE 'Credential report not ready.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when getting credential report.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte GetCredentialReport na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetGroup`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetGroup`.

CLI

AWS CLI

Obter um grupo do IAM

Este exemplo retorna detalhes sobre o grupo do IAM `Admins`.

```
aws iam get-group \
    --group-name Admins
```

Resultado:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
}
```

Para obter mais informações, consulte Identidades do IAM (usuários, grupos de usuários e perfis) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetGroup na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes sobre o grupo do IAM **`Testers`**, incluindo uma compilação de todos os usuários do IAM que pertencem ao grupo.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Saída:

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Consulte detalhes da API em GetGroup na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes sobre o grupo do IAM **`Testers`**, incluindo uma compilação de todos os usuários do IAM que pertencem ao grupo.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Saída:

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa

Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Para obter detalhes sobre a API, consulte GetGroup na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetGroupPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetGroupPolicy`.

CLI

AWS CLI

Obter informações sobre uma política anexada a um grupo do IAM

O comando `get-group-policy` a seguir obtém informações sobre a política especificada anexada ao grupo denominado `Test-Group`.

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

Resultado:

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

Para obter mais informações, consulte Gerenciamento de políticas do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes sobre a política em linha incorporada denominada **PowerUserAccess-Testers** do grupo **Testers**. A propriedade **PolicyDocument** é codificada em URL. Ela é decodificada neste exemplo com o método .NET **UrlDecode**.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Saída:

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Para ver detalhes da API, consulte GetGroupPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes sobre a política em linha incorporada denominada **`PowerUserAccess-Testers`** do grupo **`Testers`**. A propriedade **`PolicyDocument`** é codificada em URL. Ela é decodificada neste exemplo com o método .NET **`UrlDecode`**.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Saída:

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
  ]
}
```

- Para obter detalhes sobre a API, consulte GetGroupPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetInstanceProfile`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetInstanceProfile`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Conceitos básicos de streaming gerenciado

CLI

AWS CLI

Obter informações sobre um perfil de instância

O comando `get-instance-profile` a seguir obtém informações sobre o perfil de instância denominado `ExampleInstanceProfile`.

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Resultado:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
```

```
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
    }
}
```

Para obter mais informações, consulte Usar perfis de instância no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetInstanceProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes do perfil de instância denominado **`ec2instancerole`** definido na conta atual da AWS.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Saída:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Para ver detalhes da API, consulte GetInstanceProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes do perfil de instância denominado **`ec2instancerole`** definido na conta atual da AWS.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Saída:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
```

```
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Para obter detalhes sobre a API, consulte GetInstanceProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetLoginProfile`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetLoginProfile`.

CLI

AWS CLI

Obter informações de senha de um usuário do IAM

O comando `get-login-profile` a seguir obtém informações sobre a senha do usuário do IAM chamado Bob.

```
aws iam get-login-profile \
    --user-name Bob
```

Resultado:

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

O comando `get-login-profile` pode ser usado para verificar se um usuário do IAM tem uma senha. O comando retorna um erro `NoSuchEntity` se nenhuma senha for definida para o usuário.

Não é possível visualizar uma senha com esse comando. Se a senha for perdida, você pode redefini-la (`update-login-profile`) para o usuário. Como alternativa, você pode excluir o perfil de login (`delete-login-profile`) do usuário e criar um novo (`create-login-profile`).

Para obter mais informações, consulte Gerenciamento de senhas de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna a data de criação da senha e se uma redefinição de senha é necessária para o usuário do IAM **`David`**.

```
Get-IAMLoginProfile -UserName David
```

Saída:

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- Para ver detalhes da API, consulte GetLoginProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna a data de criação da senha e se uma redefinição de senha é necessária para o usuário do IAM **`David`**.

```
Get-IAMLoginProfile -UserName David
```

Saída:

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
```

```
12/10/2014 3:39:44 PM        False                                  David
```

- Para obter detalhes sobre a API, consulte GetLoginProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetOpenIdConnectProvider`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetOpenIdConnectProvider`.

CLI

AWS CLI

Retornar informações sobre o provedor OpenID Connect especificado

Este exemplo retorna detalhes sobre o provedor OpenID Connect cujo ARN é `arn:aws:iam::123456789012:oidc-provider/server.example.com`.

```
aws iam get-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Resultado:

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes sobre o provedor OpenID Connect cujo ARN é **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`**. A propriedade **`ClientIDList`** é uma compilação que contém todos os IDs de cliente definidos para esse provedor.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Saída:

```
ClientIDList         CreateDate               ThumbprintList
           Url
------------         ----------               --------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Para ver detalhes da API, consulte GetOpenIdConnectProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes sobre o provedor OpenID Connect cujo ARN é **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`**. A propriedade **`ClientIDList`** é uma compilação que contém todos os IDs de cliente definidos para esse provedor.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Saída:

```
ClientIDList         CreateDate               ThumbprintList
           Url
```

```
------------         ----------                ---------------
           ---
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Para obter detalhes sobre a API, consulte GetOpenIdConnectProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetPolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conceitos básicos de streaming gerenciado
- Trabalhar com a API IAM Policy Builder

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
```

```
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
{ PolicyArn = policyArn });
        return response.Policy;
    }
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para .NET.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
```

```
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como recuperar informações sobre a política gerenciada especificada

Este exemplo retorna detalhes sobre a política gerenciada cujo ARN é `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Resultado:

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetPolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(ctx context.Context, policyArn string)
 (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(ctx, &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
```

```
 }
 return policy, err
}
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha a política.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK for Kotlin.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
```

```
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes sobre a política gerenciada cujo ARN é **`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Saída:

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Para ver detalhes da API, consulte GetPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes sobre a política gerenciada cujo ARN é **`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Saída:

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Para obter detalhes sobre a API, consulte GetPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
```

```
            raise
        else:
            return policy_statement
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end
```

- Consulte detalhes da API em GetPolicy na Referência da API AWS SDK para Ruby.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->getpolicy( iv_policyarn = iv_policy_arn ).
        MESSAGE 'Retrieved policy information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte GetPolicy na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
```

```
        do {
            let output = try await client.getPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            print("ERROR: getPolicy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte GetPolicy na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetPolicyVersion`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetPolicyVersion`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Políticas gerenciadas
- Trabalhar com a API IAM Policy Builder

CLI

AWS CLI

Como recuperar informações sobre a versão especificada da política gerenciada especificada

Este exemplo retorna o documento da política para a versão v2 da política cujo ARN é `arn:aws:iam::123456789012:policy/MyManagedPolicy`.

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
```

```
    --version-id v2
```

Resultado:

```
{
    "PolicyVersion": {
        "Document": {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetPolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna o documento da política na versão **v2** da política cujo ARN é **arn:aws:iam::123456789012:policy/MyManagedPolicy**. O documento da política na propriedade **Document** é codificado em URL, sendo decodificado neste exemplo com o método .NET **UrlDecode**.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Saída:

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- Para ver detalhes da API, consulte GetPolicyVersion na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna o documento da política na versão **v2** da política cujo ARN é **arn:aws:iam::123456789012:policy/MyManagedPolicy**. O documento da política na propriedade **Document** é codificado em URL, sendo decodificado neste exemplo com o método .NET **UrlDecode**.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Saída:

```
CreateDate             Document
 IsDefaultVersion     VersionId
```

```
----------             --------
 ----------------     ---------
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
           v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version":"2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
      ]
    }
}
```

- Para obter detalhes sobre a API, consulte GetPolicyVersion na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
```

```
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
policy_arn)
        raise
    else:
        return policy_statement
```

- Consulte detalhes da API em GetPolicyVersion na Referência da API AWS SDK para Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->getpolicyversion(
          iv_policyarn = iv_policy_arn
          iv_versionid = iv_version_id ).
        MESSAGE 'Retrieved policy version information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Policy or version does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iaminvalidinputex.
        MESSAGE 'Invalid input provided.' TYPE 'E'.
    ENDTRY.
```

- Para obter os detalhes da API, consulte GetPolicyVersion na referência do API do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetRole`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetRole`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Configurar a conectividade do serviço de contêiner
- Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Conceitos básicos de data warehouses de tecnologia sem servidor
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos de clusters de data warehouse provisionados
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }
```

- Consulte detalhes da API em GetRole na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como obter informações sobre um perfil do IAM

O comando `get-role`, apresentado a seguir, obtém informações sobre o perfil denominado `Test-Role`.

```
aws iam get-role \
    --role-name Test-Role
```

Resultado:

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
```

```
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

O comando exibe a política de confiança anexada ao perfil. Para listar as políticas de permissões anexadas a um perfil, use o comando `list-role-policies`.

Para obter mais informações, consulte Criação de perfis do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
```

```
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(ctx context.Context, roleName string)
 (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(ctx,
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Consulte detalhes da API em GetRole na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha a função.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Consulte detalhes da API em GetRole na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Consulte detalhes da API em GetRole na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna os detalhes do **`lamda_exec_role`**. Ele inclui o documento da política de confiança que especifica quem pode assumir esse perfil. O documento da política é codificado em URL e pode ser decodificado usando o método .NET **`UrlDecode`**. Neste exemplo, todos os espaços em branco da política original foram removidos antes de ela ser carregada na política. Para ver os documentos da política de permissões que determinam o que alguém que assume o perfil pode fazer, use **`Get-IAMRolePolicy`** para políticas em linha e **`Get-IAMPolicyVersion`** para políticas gerenciadas anexadas.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Saída:

```
Arn                       : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate                : 4/2/2015 9:16:11 AM
Path                      : /
RoleId                    : 2YBIKAIBHNKB4EXAMPLE1
RoleName                  : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Saída:

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Consulte detalhes da API em GetRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna os detalhes do **`lamda_exec_role`**. Ele inclui o documento da política de confiança que especifica quem pode assumir esse perfil. O documento da política é codificado em URL e pode ser decodificado usando o método .NET **`UrlDecode`**. Neste exemplo, todos os espaços em branco da política original foram removidos antes de ela ser carregada na política. Para ver os documentos da política de permissões que determinam o que alguém que assume o perfil pode fazer, use **`Get-IAMRolePolicy`** para políticas em linha e **`Get-IAMPolicyVersion`** para políticas gerenciadas anexadas.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Saída:

```
Arn                       : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate                : 4/2/2015 9:16:11 AM
Path                      : /
RoleId                    : 2YBIKAIBHNKB4EXAMPLE1
RoleName                  : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Saída:

```
{"Version":"2012-10-17",       "Statement":
[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Para obter detalhes sobre a API, consulte GetRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Consulte detalhes da API em GetRole na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
                               role_name: name
                             }).role
  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
  role
end
```

- Consulte detalhes da API em GetRole na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Consulte detalhes da API em GetRole na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->getrole( iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved role information.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte GetRole na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func getRole(name: String) async throws -> IAMClientTypes.Role {
        let input = GetRoleInput(
            roleName: name
        )
        do {
            let output = try await client.getRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            print("ERROR: getRole:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte GetRole na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetRolePolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetRolePolicy`.

CLI

AWS CLI

Obter informações sobre uma política anexada a um perfil do IAM

O comando `get-role-policy` a seguir obtém informações sobre a política especificada anexada ao perfil denominado `Test-Role`.

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Resultado:

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

Para obter mais informações, consulte Criação de perfis do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetRolePolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna o documento da política de permissões da política denominada **oneClick_lambda_exec_role_policy** incorporada no perfil do IAM **lamda_exec_role**. O documento resultante da política é codificado em URL. Ela é decodificada neste exemplo com o método .NET **UrlDecode**.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Saída:

```
PolicyDocument                                                PolicyName
              UserName
--------------                                                ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Saída:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
```

```
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Para ver detalhes da API, consulte GetRolePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna o documento da política de permissões da política denominada **oneClick_lambda_exec_role_policy** incorporada no perfil do IAM **lamda_exec_role**. O documento resultante da política é codificado em URL. Ela é decodificada neste exemplo com o método .NET **UrlDecode**.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Saída:

```
PolicyDocument                                          PolicyName
              UserName
--------------                                          ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Saída:

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:us-east-1:555555555555:log-group:/aws/lambda/aws-
example-function:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ]
    }
  ]
}
```

- Para obter detalhes sobre a API, consulte GetRolePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetSamlProvider`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetSamlProvider`.

CLI

AWS CLI

Recuperar o metadocumento do provedor SAML

Este exemplo recupera os detalhes sobre o provedor SAML 2.0 cujo ARM é `arn:aws:iam::123456789012:saml-provider/SAMLADFS`. A resposta inclui o documento de metadados que você obteve do provedor de identidade para criar a entidade do provedor SAML da AWS, bem como as datas de criação e expiração.

```
aws iam get-saml-provider \
```

```
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Resultado:

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
    ]
}
```

Para obter mais informações, consulte Criação de provedores de identidade SAML do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetSamlProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera os detalhes sobre o provedor SAML 2.0 cujo ARM é arn:aws:iam::123456789012:saml-provider/SAMLADFS. A resposta inclui o documento de metadados que você obteve do provedor de identidade para criar a entidade do provedor SAML da AWS, bem como as datas de criação e expiração.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Saída:

```
CreateDate                SAMLMetadataDocument
        ValidUntil
```

```
----------                  --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Para ver detalhes da API, consulte GetSamlProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera os detalhes sobre o provedor SAML 2.0 cujo ARM é arn:aws:iam::123456789012:saml-provider/SAMLADFS. A resposta inclui o documento de metadados que você obteve do provedor de identidade para criar a entidade do provedor SAML da AWS, bem como as datas de criação e expiração.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Saída:

```
CreateDate                  SAMLMetadataDocument
        ValidUntil
----------                  --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Para obter detalhes sobre a API, consulte GetSamlProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetServerCertificate`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetServerCertificate`.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }
```

```
    return result;
}
```

- Consulte detalhes da API em GetServerCertificate na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como obter detalhes sobre um certificado de servidor em sua conta da AWS

O comando `get-server-certificate`, apresentado a seguir, recupera todos os detalhes sobre o certificado de servidor especificado em sua conta da AWS.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Resultado:

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
```

```
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

Para listar os certificados de servidor disponíveis em sua conta da AWS, use o comando `list-server-certificates`.

Para obter mais informações, consulte Gerenciar certificados de servidor no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetServerCertificate na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha um certificado do servidor.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em GetServerCertificate na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });
```

```
iam.getServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em GetServerCertificate na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera detalhes sobre o certificado do servidor denominado **`MyServerCertificate`**. Você pode encontrar os detalhes do certificado nas propriedades **`CertificateBody`** e **`ServerCertificateMetadata`**.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Saída:

```
CertificateBody          : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
```

```
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Saída:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Para ver detalhes da API, consulte GetServerCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera detalhes sobre o certificado do servidor denominado **MyServerCertificate**. Você pode encontrar os detalhes do certificado nas propriedades **CertificateBody** e **ServerCertificateMetadata**.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Saída:

```
CertificateBody           : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                           21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                           rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                           nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                           NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                           -----END CERTIFICATE-----
CertificateChain          :
```

```
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Saída:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Para obter detalhes sobre a API, consulte GetServerCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetServiceLastAccessedDetails`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetServiceLastAccessedDetails`.

CLI

AWS CLI

Recuperar um relatório de acesso ao serviço

O exemplo de `get-service-last-accessed-details` a seguir recupera um relatório gerado anteriormente que lista os serviços acessados pelas entidades do IAM. Para gerar um relatório, use o comando `generate-service-last-accessed-details`.

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Resultado:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

Para obter mais informações, consulte Refinar permissões na AWS usando as informações do último acesso no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetServiceLastAccessedDetails na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo fornece detalhes do último serviço acessado pela entidade do IAM (usuário, grupo, perfil ou política) associada na chamada de solicitação.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Saída:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Consulte detalhes da API em GetServiceLastAccessedDetails na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo fornece detalhes do último serviço acessado pela entidade do IAM (usuário, grupo, perfil ou política) associada na chamada de solicitação.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Saída:

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Para obter detalhes sobre a API, consulte GetServiceLastAccessedDetails na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetServiceLastAccessedDetailsWithEntities`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetServiceLastAccessedDetailsWithEntities`.

CLI

AWS CLI

Recuperar um relatório de acesso ao serviço com detalhes de um serviço

O exemplo de `get-service-last-accessed-details-with-entities` a seguir recupera um relatório que contém detalhes sobre os usuários do IAM e outras entidades que acessaram o serviço especificado. Para gerar um relatório, use o comando `generate-service-last-accessed-details`. Para obter uma lista de serviços acessados com namespaces, use `get-service-last-accessed-details`.

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

Resultado:

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

Para obter mais informações, consulte Refinar permissões na AWS usando as informações do último acesso no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetServiceLastAccessedDetailsWithEntities na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo fornece o carimbo de data e hora do último acesso do serviço na solicitação pela respectiva entidade do IAM.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Saída:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Saída:

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Saída:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Consulte detalhes da API em GetServiceLastAccessedDetailsWithEntities na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo fornece o carimbo de data e hora do último acesso do serviço na solicitação pela respectiva entidade do IAM.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Saída:

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Saída:

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Saída:

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
Type : USER
```

- Para obter detalhes sobre a API, consulte GetServiceLastAccessedDetailsWithEntities na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **GetServiceLinkedRoleDeletionStatus** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetServiceLinkedRoleDeletionStatus`.

CLI

AWS CLI

Como verificar o status de uma solicitação para excluir um perfil vinculado ao serviço

O exemplo de `get-service-linked-role-deletion-status`, apresentado a seguir, exibe o status de uma solicitação anterior para excluir um perfil vinculado ao serviço. A operação de exclusão ocorre de forma assíncrona. Ao fazer a solicitação, você obtém um valor `DeletionTaskId` fornecido como parâmetro para esse comando.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Resultado:

```
{
"Status": "SUCCEEDED"
}
```

Para obter mais informações, consulte Usar perfis vinculados ao serviço no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetServiceLinkedRoleDeletionStatus na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Para obter os detalhes da API, consulte GetServiceLinkedRoleDeletionStatus na Referência de API do AWS SDK para JavaScript.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetUser`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetUser`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Consulte detalhes da API em GetUser na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
```

```
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Consulte detalhes da API em GetUser na Referência de comandos da AWS CLI.

CLI

AWS CLI

Como obter informações sobre um usuário do IAM

O comando `get-user`, apresentado a seguir, obtém informações sobre o usuário do IAM denominado `Paulo`.

```
aws iam get-user \
    --user-name Paulo
```

Resultado:

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

Para obter mais informações, consulte Gerenciar usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetUser na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
```

```
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(ctx context.Context, userName string)
 (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(ctx, &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

- Consulte detalhes da API em GetUser na Referência da API do AWS SDK para Go.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera detalhes sobre o usuário chamado **David**.

```
Get-IAMUser -UserName David
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Exemplo 2: este exemplo recupera detalhes sobre o usuário do IAM atualmente conectado.

```
Get-IAMUser
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Consulte detalhes da API em GetUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera detalhes sobre o usuário chamado **`David`**.

```
Get-IAMUser -UserName David
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
```

```
UserName         : David
```

Exemplo 2: este exemplo recupera detalhes sobre o usuário do IAM atualmente conectado.

```
Get-IAMUser
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Para obter detalhes sobre a API, consulte GetUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  # Retrieves a user's details
  #
  # @param user_name [String] The name of the user to retrieve
  # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
  def get_user(user_name)
    response = @iam_client.get_user(user_name: user_name)
    response.user
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("User '#{user_name}' not found.")
    nil
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error retrieving user '#{user_name}': #{e.message}")
    nil
  end
```

- Consulte detalhes da API em GetUser na Referência da API do AWS SDK para Ruby.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func getUser(name: String? = nil) async throws -> IAMClientTypes.User
 {
        let input = GetUserInput(
            userName: name
        )
        do {
            let output = try await iamClient.getUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            return user
        } catch {
            print("ERROR: getUser:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes sobre a API, consulte GetUser na AWS SDK for Swift API Reference.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetUserPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetUserPolicy`.

CLI

AWS CLI

Listar detalhes da política de um usuário do IAM

O comando `get-user-policy` a seguir lista os detalhes da política especificada anexada ao usuário do IAM chamado Bob.

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Resultado:

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
}
```

Para obter uma lista de políticas para um usuário do IAM, use o comando `list-user-policies`.

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetUserPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera os detalhes da política em linha denominada **Davids_IAM_Admin_Policy** incorporada no usuário do IAM chamado **David**. O documento de política é codificado em URL.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Saída:

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
  ]
}
```

- Para ver detalhes da API, consulte GetUserPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera os detalhes da política em linha denominada **`Davids_IAM_Admin_Policy`** incorporada no usuário do IAM chamado **`David`**. O documento de política é codificado em URL.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Saída:

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:user/*"
      ]
    }
  ]
}
```

- Para obter detalhes sobre a API, consulte GetUserPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAccessKeys`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListAccessKeys`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar chaves de acesso

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
```

```
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
    --output text \
```

```
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em ListAccessKeys na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
```

```
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK para C++.

CLI

AWS CLI

Para listar os IDs da chave de acesso para um usuário do IAM

O comando `list-access-keys`, apresentado a seguir, lista os IDs das chaves de acesso para o usuário do IAM denominado Bob.

```
aws iam list-access-keys \
    --user-name Bob
```

Resultado:

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

Não é possível listar as chaves de acesso secretas para os usuários do IAM. Se as chaves de acesso secretas forem perdidas, você deverá criar novas chaves de acesso usando o comando `create-access-keys`.

Para obter mais informações, consulte Gerenciamento de chaves de acesso de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListAccessKeys na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(ctx context.Context, userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(ctx, &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
```

```
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s
```

```
                Where:
                    userName - The name of the user for which access keys are
retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();

                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
                            .build();
```

```
                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
 metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste as chaves de acesso.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
```

```
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
            userName = userNameVal
        }
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando lista as chaves de acesso do usuário do IAM chamado **Bob**. Observe que não é possível listar as chaves de acesso secretas dos usuários do IAM. Se as chaves de acesso secretas forem perdidas, você deverá criar novas chaves de acesso com o cmdlet **New-IAMAccessKey**.

```
Get-IAMAccessKey -UserName "Bob"
```

Saída:

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Para ver detalhes da API, consulte ListAccessKeys na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando lista as chaves de acesso do usuário do IAM chamado **Bob**. Observe que não é possível listar as chaves de acesso secretas dos usuários do IAM. Se as chaves de acesso secretas forem perdidas, você deverá criar novas chaves de acesso com o cmdlet **New-IAMAccessKey**.

```
Get-IAMAccessKey -UserName "Bob"
```

Saída:

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Para obter detalhes sobre a API, consulte ListAccessKeys na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, desativa e exclui chaves de acesso.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'AccessKeyManager'
  end
```

```
  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded
    @logger.error('Error creating access key: limit exceeded. Cannot create
more.')
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
```

```
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: 'Inactive'
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em ListAccessKeys na Referência da API AWS SDK para Ruby.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
```

```
        oo_result = lo_iam->listaccesskeys(
          iv_username = iv_user_name ).
        MESSAGE 'Retrieved access key list.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte ListAccessKeys na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAccountAliases`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListAccountAliases`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar sua conta

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;
```

```
    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Consulte detalhes da API em ListAccountAliases na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como listar os aliases da conta

O comando `list-account-aliases`, apresentado a seguir, lista os aliases para a conta atual.

```
aws iam list-account-aliases
```

Resultado:

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

Para obter mais informações, consulte O ID da sua conta da AWS e seu alias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListAccountAliases na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
```

```
 */
public class ListAccountAliases {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em ListAccountAliases na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste os aliases de conta.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListAccountAliases na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListAccountAliases na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun listAliases() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- Consulte detalhes da API em ListAccountAliases na Referência da API AWS SDK for Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando retorna o alias da conta da Conta da AWS.

```
Get-IAMAccountAlias
```

Saída:

```
ExampleCo
```

- Para ver detalhes da API, consulte ListAccountAliases na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando retorna o alias da conta da Conta da AWS.

```
Get-IAMAccountAlias
```

Saída:

```
ExampleCo
```

- Para obter detalhes sobre a API, consulte ListAccountAliases na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

- Consulte detalhes da API em ListAccountAliases, na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Listar, criar e excluir aliases da conta.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info('Account aliases are:')
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info('No account aliases found.')
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em ListAccountAliases na Referência da API AWS SDK para Ruby.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->listaccountaliases( ).
        MESSAGE 'Retrieved account alias list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing account aliases.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte ListAccountAliases na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAttachedGroupPolicies`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListAttachedGroupPolicies`.

CLI

AWS CLI

Para listar todas as políticas gerenciadas anexadas ao grupo especificado

Este exemplo retorna os nomes e os ARNs das políticas gerenciadas anexadas ao grupo do IAM denominado `Admins` na conta da AWS.

```
aws iam list-attached-group-policies \
    --group-name Admins
```

Resultado:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListAttachedGroupPolicies na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando retorna os nomes e os ARNs das políticas gerenciadas anexadas ao grupo do IAM denominado **Admins** na conta da AWS. Para ver a lista de políticas em linha incorporadas no grupo, use o comando **Get-IAMGroupPolicyList**.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Saída:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- Para ver detalhes da API, consulte ListAttachedGroupPolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando retorna os nomes e os ARNs das políticas gerenciadas anexadas ao grupo do IAM denominado **Admins** na conta da AWS. Para ver a lista de políticas em linha incorporadas no grupo, use o comando **Get-IAMGroupPolicyList**.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Saída:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess               AdministratorAccess
```

- Para obter detalhes sobre a API, consulte ListAttachedGroupPolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAttachedRolePolicies`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListAttachedRolePolicies`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
   /// <summary>
   /// List the IAM role policies that are attached to an IAM role.
   /// </summary>
   /// <param name="roleName">The IAM role to list IAM policies for.</param>
   /// <returns>A list of the IAM policies attached to the IAM role.</returns>
   public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
   {
       var attachedPolicies = new List<AttachedPolicyType>();
       var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

       await foreach (var response in attachedRolePoliciesPaginator.Responses)
       {
           attachedPolicies.AddRange(response.AttachedPolicies);
       }

       return attachedPolicies;
   }
```

- Consulte detalhes da API em ListAttachedRolePolicies na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como listar todas as políticas gerenciadas anexadas ao perfil especificado

Este comando retorna os nomes e os ARNs das políticas gerenciadas anexadas ao perfil do IAM denominado `SecurityAuditRole` na conta da AWS.

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Resultado:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListAttachedRolePolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(ctx context.Context, roleName
 string) ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(ctx,
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
```

```
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- Consulte detalhes da API em ListAttachedRolePolicies na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Lista as políticas que estão anexadas a uma função.

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });
```

```
  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Consulte detalhes da API em ListAttachedRolePolicies na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
```

```
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Consulte detalhes da API em ListAttachedRolePolicies na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este comando retorna os nomes e os ARNs das políticas gerenciadas anexadas ao perfil do IAM denominado **SecurityAuditRole** na conta da AWS. Para ver a lista de políticas em linha incorporadas no perfil, use o comando **Get-IAMRolePolicyList**.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Saída:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Para ver detalhes da API, consulte ListAttachedRolePolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este comando retorna os nomes e os ARNs das políticas gerenciadas anexadas ao perfil do IAM denominado **SecurityAuditRole** na conta da AWS. Para ver a lista de políticas em linha incorporadas no perfil, use o comando **Get-IAMRolePolicyList**.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Saída:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Para obter detalhes sobre a API, consulte ListAttachedRolePolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Consulte detalhes da API em ListAttachedRolePolicies, na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, anexa e desconecta políticas de perfis.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em ListAttachedRolePolicies na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_attached_role_policies(
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
```

```
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Consulte detalhes da API em ListAttachedRolePolicies na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->listattachedrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved attached policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3



    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
 [IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []

        // Use "Paginated" to get all the attached role polices.
        // This lets the SDK handle the 'isTruncated' in
 "ListAttachedRolePoliciesOutput".
        let input = ListAttachedRolePoliciesInput(
            roleName: role
        )
        let output = client.listAttachedRolePoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let attachedPolicies = page.attachedPolicies else {
                    print("Error: no attached policies returned.")
                    continue
                }
                for attachedPolicy in attachedPolicies {
                    policyList.append(attachedPolicy)
```

```
                }
            }
        } catch {
            print("ERROR: listAttachedRolePolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAttachedUserPolicies`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListAttachedUserPolicies`.

CLI

AWS CLI

Para listar todas as políticas gerenciadas anexadas ao usuário especificado

Este comando retorna os nomes e os ARNs das políticas gerenciadas do usuário do IAM chamado Bob na conta da AWS.

```
aws iam list-attached-user-policies \
    --user-name Bob
```

Resultado:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
```

```
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListAttachedUserPolicies na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo1: este comando retorna os nomes e os ARNs das políticas gerenciadas do usuário do IAM chamado **Bob** na conta da AWS. Para ver a lista de políticas em linha incorporadas no usuário do IAM, use o comando **Get-IAMUserPolicyList**.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Saída:

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/TesterPolicy                     TesterPolicy
```

- Para ver detalhes da API, consulte ListAttachedUserPolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo1: este comando retorna os nomes e os ARNs das políticas gerenciadas do usuário do IAM chamado **Bob** na conta da AWS. Para ver a lista de políticas em linha incorporadas no usuário do IAM, use o comando **Get-IAMUserPolicyList**.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Saída:

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/TesterPolicy                      TesterPolicy
```

- Para obter detalhes sobre a API, consulte ListAttachedUserPolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListEntitiesForPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListEntitiesForPolicy`.

CLI

AWS CLI

Para listar todos os usuários, grupos e perfis aos quais a política gerenciada especificada está anexada

Este exemplo retorna uma lista de grupos, perfis e usuários do IAM que têm a política `arn:aws:iam::123456789012:policy/TestPolicy` anexada.

```
aws iam list-entities-for-policy \
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Resultado:

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
```

```
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListEntitiesForPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna uma lista de grupos, perfis e usuários do IAM que têm a política **`arn:aws:iam::123456789012:policy/TestPolicy`** anexada.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Saída:

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Para ver detalhes da API, consulte ListEntitiesForPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna uma lista de grupos, perfis e usuários do IAM que têm a política **`arn:aws:iam::123456789012:policy/TestPolicy`** anexada.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Saída:

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Para obter detalhes sobre a API, consulte ListEntitiesForPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListGroupPolicies`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListGroupPolicies`.

CLI

AWS CLI

Para listar todas as políticas em linha anexadas ao grupo especificado

O comando `list-group-policies` a seguir lista os nomes das políticas em linha anexadas ao grupo do IAM denominado `Admins` na conta atual.

```
aws iam list-group-policies \
    --group-name Admins
```

Resultado:

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
```

```
}
```

Para obter mais informações, consulte Gerenciamento de políticas do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListGroupPolicies na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna uma lista das políticas em linha incorporadas no grupo **`Testers`**. Para obter as políticas gerenciadas anexadas ao grupo, use o comando **`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Saída:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Para ver detalhes da API, consulte ListGroupPolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna uma lista das políticas em linha incorporadas no grupo **`Testers`**. Para obter as políticas gerenciadas anexadas ao grupo, use o comando **`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Saída:

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Para obter detalhes sobre a API, consulte ListGroupPolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListGroups`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListGroups`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Consulte detalhes da API em ListGroups na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como listar os grupos do IAM para a conta atual

O comando `list-groups`, apresentado a seguir, lista os grupos do IAM na conta atual.

```
aws iam list-groups
```

Resultado:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Para obter mais informações, consulte Gerenciar grupos de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListGroups na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(ctx context.Context, maxGroups int32)
 ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(ctx, &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
```

```
 return groups, err
}
```

- Consulte detalhes da API em ListGroups na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste os grupos.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
```

```
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Consulte detalhes da API em ListGroups na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }
```

```
        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- Consulte detalhes da API em ListGroups na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna uma compilação de todos os grupos do IAM definidos na atual Conta da AWS.

```
Get-IAMGroupList
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Consulte detalhes da API em ListGroups na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna uma compilação de todos os grupos do IAM definidos na atual Conta da AWS.

```
Get-IAMGroupList
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Para obter detalhes sobre a API, consulte ListGroups na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_groups(count):
```

```
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Consulte detalhes da API em ListGroups na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
```

```
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Consulte detalhes da API em ListGroups na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Consulte detalhes da API em ListGroups na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->listgroups( ).
    MESSAGE 'Retrieved group list.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing groups.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte ListGroups na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3
```

```
    public func listGroups() async throws -> [String] {
        var groupList: [String] = []

        // Use "Paginated" to get all the groups.
        // This lets the SDK handle the 'isTruncated' property in
"ListGroupsOutput".
        let input = ListGroupsInput()

        let pages = client.listGroupsPaginated(input: input)
        do {
            for try await page in pages {
                guard let groups = page.groups else {
                    print("Error: no groups returned.")
                    continue
                }

                for group in groups {
                    if let name = group.groupName {
                        groupList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listGroups:", dump(error))
            throw error
        }
        return groupList
    }
```

- Para obter detalhes da API, consulte ListGroups na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListGroupsForUser`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListGroupsForUser`.

CLI

AWS CLI

Listar os grupos aos quais um usuário do IAM pertence

O comando `list-groups-for-user` a seguir exibe os grupos aos quais o usuário do IAM chamado Bob pertence.

```
aws iam list-groups-for-user \
    --user-name Bob
```

Resultado:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

Para obter mais informações, consulte Gerenciar grupos de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListGroupsForUser na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna a lista de grupos do IAM aos quais o usuário do IAM **David** pertence.

```
Get-IAMGroupForUser -UserName David
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Para ver detalhes da API, consulte ListGroupsForUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna a lista de grupos do IAM aos quais o usuário do IAM **David** pertence.

```
Get-IAMGroupForUser -UserName David
```

Saída:

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
```

```
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
GroupName  : Developers
Path       : /
```

- Para obter detalhes sobre a API, consulte ListGroupsForUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListInstanceProfiles`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListInstanceProfiles`.

CLI

AWS CLI

Listar os perfis de instância da conta

O comando `list-instance-profiles` a seguir lista os perfis de instância associados à conta atual.

```
aws iam list-instance-profiles
```

Resultado:

```
{
    "InstanceProfiles": [
```

```
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version":"2012-10-17",
                        "Statement": [
```

```
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
}
```

Para obter mais informações, consulte Usar perfis de instância no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListInstanceProfiles na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna uma compilação dos perfis de instância definidos na atual Conta da AWS.

```
Get-IAMInstanceProfileList
```

Saída:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Consulte detalhes da API em ListInstanceProfiles na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna uma compilação dos perfis de instância definidos na atual Conta da AWS.

```
Get-IAMInstanceProfileList
```

Saída:

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Para obter detalhes sobre a API, consulte ListInstanceProfiles na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListInstanceProfilesForRole`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListInstanceProfilesForRole`.

CLI

AWS CLI

Listar os perfis de instância de um perfil do IAM

O comando `list-instance-profiles-for-role` a seguir lista os perfis de instância associados ao perfil `Test-Role`.

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

Resultado:

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

Para obter mais informações, consulte Usar perfis de instância no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListInstanceProfilesForRole na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes do perfil de instância associado ao perfil **`ec2instancerole`**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Saída:

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
```

```
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Consulte detalhes da API em ListInstanceProfilesForRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes do perfil de instância associado ao perfil **`ec2instancerole`**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Saída:

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Para obter detalhes sobre a API, consulte ListInstanceProfilesForRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListMfaDevices`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListMfaDevices`.

CLI

AWS CLI

Listar todos os dispositivos de MFA de um usuário especificado

Este exemplo retorna detalhes sobre o dispositivo de MFA atribuído ao usuário do IAM Bob.

```
aws iam list-mfa-devices \
    --user-name Bob
```

Resultado:

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

Para obter mais informações, consulte Uso de autenticação multifator (MFA) na AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListMfaDevices na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna detalhes sobre o dispositivo de MFA atribuído ao usuário do IAM **David**. Neste exemplo, você percebe que é um dispositivo virtual porque o **SerialNumber** é um ARN em vez do número de série real de um dispositivo físico.

```
Get-IAMMFADevice -UserName David
```

Saída:

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- Para ver detalhes da API, consulte ListMfaDevices na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna detalhes sobre o dispositivo de MFA atribuído ao usuário do IAM **David**. Neste exemplo, você percebe que é um dispositivo virtual porque o **SerialNumber** é um ARN em vez do número de série real de um dispositivo físico.

```
Get-IAMMFADevice -UserName David
```

Saída:

```
EnableDate                SerialNumber                            UserName
----------                ------------                            --------
4/8/2015 9:41:10 AM       arn:aws:iam::123456789012:mfa/David     David
```

- Para obter detalhes sobre a API, consulte ListMfaDevices na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListOpenIdConnectProviders`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListOpenIdConnectProviders`.

CLI

AWS CLI

Para listar informações sobre os provedores OpenID Connect na conta da AWS

Este exemplo retorna uma lista de ARNS de todos os provedores OpenID Connect definidos na conta atual da AWS.

```
aws iam list-open-id-connect-providers
```

Resultado:

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
    ]
}
```

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListOpenIdConnectProviders na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna uma lista de ARNS de todos os provedores OpenID Connect definidos na atual Conta da AWS.

```
Get-IAMOpenIDConnectProviderList
```

Saída:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Para ver detalhes da API, consulte ListOpenIdConnectProviders na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna uma lista de ARNS de todos os provedores OpenID Connect definidos na atual Conta da AWS.

```
Get-IAMOpenIDConnectProviderList
```

Saída:

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Para obter detalhes sobre a API, consulte ListOpenIdConnectProviders na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListPolicies`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListPolicies`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Políticas gerenciadas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para .NET.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }
```

```
    return true;
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para C++.

CLI

AWS CLI

Para listar as políticas gerenciadas disponíveis para sua conta da AWS

Este exemplo retorna uma compilação das duas primeiras políticas gerenciadas disponíveis na conta da AWS atual.

```
aws iam list-policies \
    --max-items 3
```

Resultado:

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
```

```
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListPolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
```

```
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(ctx context.Context, maxPolicies int32)
 ([]types.Policy, error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(ctx, &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste as políticas.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Consulte detalhes da API em ListPolicies na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna uma compilação das três primeiras políticas gerenciadas disponíveis na conta atual da AWS. Como o **`-scope`** não está especificado, ele usa **`all`** como padrão e inclui políticas gerenciadas pelo cliente e pela AWS.

```
Get-IAMPolicyList -MaxItem 3
```

Saída:

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
```

```
UpdateDate       : 2/11/2015 9:21:45 AM
```

Exemplo 2: este exemplo retorna uma compilação das duas primeiras políticas gerenciadas pelo cliente disponíveis na conta atual da AWS. Ele usa **`-Scope local`** para limitar a saída somente às políticas gerenciadas pelo cliente.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Saída:

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Para ver detalhes da API, consulte ListPolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna uma compilação das três primeiras políticas gerenciadas disponíveis na conta atual da AWS. Como o **`-scope`** não está especificado, ele usa **`all`** como padrão e inclui políticas gerenciadas pelo cliente e pela AWS.

```
Get-IAMPolicyList -MaxItem 3
```

Saída:

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Exemplo 2: este exemplo retorna uma compilação das duas primeiras políticas gerenciadas pelo cliente disponíveis na conta atual da AWS. Ele usa **`-Scope local`** para limitar a saída somente às políticas gerenciadas pelo cliente.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Saída:

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Para obter detalhes sobre a API, consulte ListPolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
```

```
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Consulte detalhes da API em ListPolicies, na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Este exemplo de módulo lista, cria, anexa e desconecta políticas de perfis.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'PolicyManager'
  end
```

```
  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
```

```
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em ListPolicies na Referência da API do AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Consulte detalhes da API em ListPolicies na Referência da API AWS SDK para Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->listpolicies( iv_scope = iv_scope ).
        MESSAGE 'Retrieved policy list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing policies.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte ListPolicies na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []

        // Use "Paginated" to get all the policies.
```

```
        // This lets the SDK handle the 'isTruncated' in "ListPoliciesOutput".
        let input = ListPoliciesInput()
        let output = client.listPoliciesPaginated(input: input)

        do {
            for try await page in output {
                guard let policies = page.policies else {
                    print("Error: no policies returned.")
                    continue
                }

                for policy in policies {
                    guard let name = policy.policyName,
                          let id = policy.policyId,
                          let arn = policy.arn
                    else {
                        throw ServiceHandlerError.noSuchPolicy
                    }
                    policyList.append(MyPolicyRecord(name: name, id: id, arn:
arn))
                }
            }
        } catch {
            print("ERROR: listPolicies:", dump(error))
            throw error
        }

        return policyList
    }
```

- Para obter detalhes da API, consulte ListPolicies na referência do AWS SDK para API Swift API.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListPolicyVersions`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListPolicyVersions`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Políticas gerenciadas
- Reverter uma versão de política

CLI

AWS CLI

Listar informações sobre as versões da política gerenciada especificada

Este exemplo retorna a lista de versões disponíveis da política cujo ARN é `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Resultado:

```
{
    "IsTruncated": false,
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListPolicyVersions na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna a lista de versões disponíveis da política cujo ARN é **`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Para obter o documento de política de uma versão específica, use o comando **`Get-IAMPolicyVersion`** e especifique o **`VersionId`** do que você deseja.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Saída:

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
2/12/2015 9:39:53 AM                                True
       v2
2/12/2015 9:39:09 AM                                False
       v1
```

- Para ver detalhes da API, consulte ListPolicyVersions na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna a lista de versões disponíveis da política cujo ARN é **`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Para obter o documento de política de uma versão específica, use o comando **`Get-IAMPolicyVersion`** e especifique o **`VersionId`** do que você deseja.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Saída:

```
CreateDate                  Document                IsDefaultVersion
       VersionId
----------                  --------                ----------------
       ---------
```

```
2/12/2015 9:39:53 AM                                    True
       v2
2/12/2015 9:39:09 AM                                    False
       v1
```

- Para obter detalhes sobre a API, consulte ListPolicyVersions na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->listpolicyversions(
      iv_policyarn = iv_policy_arn ).
    MESSAGE 'Retrieved policy versions list.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing policy versions.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte ListPolicyVersions na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListRolePolicies`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListRolePolicies`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Conceitos básicos da proteção de dispositivos da Internet das Coisas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Consulte detalhes da API em ListRolePolicies na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como listar as políticas anexadas a um perfil do IAM

O comando `list-role-policies`, apresentado a seguir, lista os nomes das políticas de permissões para o perfil do IAM especificado.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Resultado:

```
{
    "PolicyNames": [
        "ExamplePolicy"
    ]
}
```

Para visualizar a política de confiança anexada a um perfil, use o comando `get-role`. Para visualizar os detalhes de uma política de permissões, use o comando `get-role-policy`.

Para obter mais informações, consulte Criação de perfis do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListRolePolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(ctx context.Context, roleName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(ctx,
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Consulte detalhes da API em ListRolePolicies na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste as políticas.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
```

```
        }),
      );
    } else {
      break;
    }
  }
}
```

- Consulte detalhes da API em ListRolePolicies na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- Consulte detalhes da API em ListRolePolicies na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo retorna a lista de nomes de políticas em linha incorporadas no perfil do IAM **`lamda_exec_role`**. Para ver os detalhes de uma política em linha, use o comando **`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Saída:

```
oneClick_lambda_exec_role_policy
```

- Para ver detalhes da API, consulte ListRolePolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo retorna a lista de nomes de políticas em linha incorporadas no perfil do IAM **`lamda_exec_role`**. Para ver os detalhes de uma política em linha, use o comando **`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Saída:

```
oneClick_lambda_exec_role_policy
```

- Para obter detalhes sobre a API, consulte ListRolePolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- Consulte detalhes da API em ListRolePolicies na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end
```

- Consulte detalhes da API em ListRolePolicies na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Consulte detalhes da API em ListRolePolicies na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->listrolepolicies(
          iv_rolename = iv_role_name ).
        MESSAGE 'Retrieved inline policy list for role.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Role does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3
```

```
    public func listRolePolicies(role: String) async throws -> [String] {
        var policyList: [String] = []

        // Use "Paginated" to get all the role policies.
        // This lets the SDK handle the 'isTruncated' in
"ListRolePoliciesOutput".
        let input = ListRolePoliciesInput(
            roleName: role
        )
        let pages = client.listRolePoliciesPaginated(input: input)

        do {
            for try await page in pages {
                guard let policies = page.policyNames else {
                    print("Error: no role policies returned.")
                    continue
                }

                for policy in policies {
                    policyList.append(policy)
                }
            }
        } catch {
            print("ERROR: listRolePolicies:", dump(error))
            throw error
        }
        return policyList
    }
```

- Para obter detalhes da API, consulte ListRolePolicies na referência do AWS SDK para API Swift API.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListRoleTags`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListRoleTags`.

CLI

AWS CLI

Listar as tags anexadas a um perfil

O comando `list-role-tags` a seguir recupera a lista de tags associadas ao perfil especificado.

```
aws iam list-role-tags \
    --role-name production-role
```

Resultado:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Para obter mais informações, consulte Recursos de tags do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListRoleTags na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo busca a tag associada ao perfil.

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Para ver detalhes da API, consulte ListRoleTags na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo busca a tag associada ao perfil..

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Para obter detalhes sobre a API, consulte ListRoleTags na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListRoles`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListRoles`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
```

```
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- Consulte detalhes da API em ListRoles na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Como listar os perfis do IAM para a conta atual

O comando `list-roles`, apresentado a seguir, lista os perfis do IAM para a conta atual.

```
aws iam list-roles
```

Resultado:

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
```

```
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Para obter mais informações, consulte Criação de perfis do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListRoles na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(ctx context.Context, maxRoles int32)
 ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(ctx,
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Consulte detalhes da API em ListRoles na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste os perfis.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
```

```
      );
    } else {
      break;
    }
  }
}
```

- Consulte detalhes da API em ListRoles na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listRolesArguments["MaxItems"] = $maxItems;
        }
```

```
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Consulte detalhes da API em ListRoles na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera uma lista de todos os perfis do IAM na Conta da AWS.

```
Get-IAMRoleList
```

- Consulte detalhes da API em ListRoles na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera uma lista de todos os perfis do IAM na Conta da AWS.

```
Get-IAMRoleList
```

- Para obter detalhes sobre a API, consulte ListRoles na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.
```

```
    :param count: The number of roles to list.
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Consulte detalhes da API em ListFunctions, na Referência da API do AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count

      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
```

```
        roles_counted += 1
      end
      break if roles_counted >= count
    end

    role_names
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't list roles for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Consulte detalhes da API em ListRoles na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Consulte detalhes da API em ListRoles na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        oo_result = lo_iam->listroles( ).
        MESSAGE 'Retrieved role list.' TYPE 'I'.
      CATCH /aws1/cx_iamservicefailureex.
        MESSAGE 'Service failure when listing roles.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte ListRoles na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    public func listRoles() async throws -> [String] {
        var roleList: [String] = []
```

```
        // Use "Paginated" to get all the roles.
        // This lets the SDK handle the 'isTruncated' in "ListRolesOutput".
        let input = ListRolesInput()
        let pages = client.listRolesPaginated(input: input)

        do {
            for try await page in pages {
                guard let roles = page.roles else {
                    print("Error: no roles returned.")
                    continue
                }

                for role in roles {
                    if let name = role.roleName {
                        roleList.append(name)
                    }
                }
            }
        } catch {
            print("ERROR: listRoles:", dump(error))
            throw error
        }
        return roleList
    }
```

- Para obter detalhes da API, consulte ListRoles na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListSAMLProviders`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListSAMLProviders`.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Consulte detalhes da API em ListSAMLProvider na Referência da API AWS SDK para .NET.

CLI

AWS CLI

Para listar os provedores SAML na conta da AWS

Este exemplo recupera a lista de provedores SAML 2.0 criados na conta da AWS atual.

```
aws iam list-saml-providers
```

Resultado:

```
{
    "SAMLProviderList": [
        {
```

```
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Para obter mais informações, consulte Criação de provedores de identidade SAML do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListSAMLProviders na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
)

// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
```

```
func (wrapper AccountWrapper) ListSAMLProviders(ctx context.Context)
 ([]types.SAMLProviderListEntry, error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(ctx,
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Consulte detalhes da API em ListSAMLProvider na Referência da API AWS SDK para Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste os provedores SAML.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Consulte detalhes da API em ListSAMLProvider na Referência da API do AWS SDK para JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Consulte detalhes da API em ListSAMLProvider na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera a lista de provedores SAML 2.0 criados na atual Conta da AWS. Ele retorna o ARN, a data de criação e a data de expiração de cada provedor SAML.

```
Get-IAMSAMLProviderList
```

Saída:

```
Arn                                                 CreateDate
   ValidUntil
---                                                 ----------
   ----------
```

```
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Para ver detalhes da API, consulte ListSAMLProviders na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera a lista de provedores SAML 2.0 criados na atual Conta da AWS. Ele retorna o ARN, a data de criação e a data de expiração de cada provedor SAML.

```
Get-IAMSAMLProviderList
```

Saída:

```
Arn                                                 CreateDate
   ValidUntil
---                                                 ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Para obter detalhes sobre a API, consulte ListSAMLProviders na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
```

```
    try:
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Consulte detalhes da API em ListSAMLProvider na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
```

```
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Consulte detalhes da API em ListSAMLProvider na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Consulte detalhes da API em ListSAMLProvider na referência da API do AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->listsamlproviders( ).
    MESSAGE 'Retrieved SAML provider list.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing SAML providers.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte ListSAMLProviders na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListServerCertificates`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListServerCertificates`.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::listServerCertificates(
```

```
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
                      certificate.GetArn() << std::setw(14) <<

certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::setw(14) <<

certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
```

```
            done = true;
        }
    }

    return true;
}
```

- Consulte detalhes da API em ListServerCertificates na Referência da API AWS SDK para C ++.

CLI

AWS CLI

Para listar os certificados de servidor em sua conta da AWS

O comando `list-server-certificates`, apresentado a seguir, lista todos os certificados de servidor armazenados e disponíveis para uso em sua conta da AWS.

```
aws iam list-server-certificates
```

Resultado:

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
```

```
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

Para obter mais informações, consulte Gerenciar certificados de servidor no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListServerCertificates na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste os certificados.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }
```

```
    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListServerCertificates na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.

- Consulte detalhes da API em ListServerCertificates na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera a lista de certificados de servidor enviados à atual Conta da AWS.

```
Get-IAMServerCertificateList
```

Saída:

```
Arn                  : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration           : 1/14/2018 9:52:36 AM
Path                 : /Org1/Org2/
ServerCertificateId  : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate           : 4/21/2015 11:14:16 AM
```

- Para ver detalhes da API, consulte ListServerCertificates na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera a lista de certificados de servidor enviados à atual Conta da AWS.

```
Get-IAMServerCertificateList
```

Saída:

```
Arn                  : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration           : 1/14/2018 9:52:36 AM
Path                 : /Org1/Org2/
ServerCertificateId  : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate           : 4/21/2015 11:14:16 AM
```

- Para obter detalhes sobre a API, consulte ListServerCertificates na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Listar, atualizar e excluir certificados de servidor.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates
```

```
    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em ListServerCertificates na Referência da API AWS SDK para Ruby.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListSigningCertificates`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListSigningCertificates`.

CLI

AWS CLI

Listar os certificados de assinatura de um usuário do IAM

O comando `list-signing-certificates` a seguir lista os certificados de assinatura do usuário do IAM denominado Bob.

```
aws iam list-signing-certificates \
    --user-name Bob
```

Resultado:

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

Para obter mais informações, consulte Manage signing certificates no Guia do usuário do Amazon EC2.

- Consulte detalhes da API em ListSigningCertificates na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera detalhes sobre o certificado de assinatura associado ao usuário chamado **Bob**.

```
Get-IAMSigningCertificate -UserName Bob
```

Saída:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
```

```
UserName        : Bob
```

- Para ver detalhes da API, consulte ListSigningCertificates na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera detalhes sobre o certificado de assinatura associado ao usuário chamado **Bob**.

```
Get-IAMSigningCertificate -UserName Bob
```

Saída:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
```

```
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Para obter detalhes sobre a API, consulte ListSignnagCertificates na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListUserPolicies`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListUserPolicies`.

CLI

AWS CLI

Para listar as políticas para um usuário do IAM

O comando `list-user-policies`, apresentado a seguir, lista as políticas anexadas ao usuário do IAM denominado Bob.

```
aws iam list-user-policies \
    --user-name Bob
```

Resultado:

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Para obter mais informações, consulte Criar um usuário do IAM na sua conta da AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListUserPolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(ctx context.Context, userName string)
 ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(ctx,
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
```

```
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Consulte detalhes da API em ListUserPolicies na Referência da API do AWS SDK para Go.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera a lista de nomes das políticas em linha incorporadas no usuário do IAM chamado **David**.

```
Get-IAMUserPolicyList -UserName David
```

Saída:

```
Davids_IAM_Admin_Policy
```

- Para ver detalhes da API, consulte ListUserPolicies na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera a lista de nomes das políticas em linha incorporadas no usuário do IAM chamado **David**.

```
Get-IAMUserPolicyList -UserName David
```

Saída:

```
Davids_IAM_Admin_Policy
```

- Para obter detalhes sobre a API, consulte ListUserPolicies na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListUserTags`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListUserTags`.

CLI

AWS CLI

Listar as tags anexadas a um usuário

O comando `list-user-tags` a seguir recupera a lista de tags associadas ao usuário do IAM especificado.

```
aws iam list-user-tags \
    --user-name alice
```

Resultado:

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Para obter mais informações, consulte Recursos de tags do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListUserTags na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo busca a tag associada ao usuário.

```
Get-IAMUserTagList -UserName joe
```

- Para ver detalhes da API, consulte ListUserTags na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo busca a tag associada ao usuário.

```
Get-IAMUserTagList -UserName joe
```

- Para obter detalhes sobre a API, consulte ListUserTags na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListUsers`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `ListUsers`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Criar usuários somente leitura e leitura/gravação usando

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
########################################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em ListUsers na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
```

```
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para C++.

CLI

AWS CLI

Para listar os usuários do IAM

O comando `list-users`, apresentado a seguir, lista os usuários do IAM na conta atual.

```
aws iam list-users
```

Resultado:

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
}
```

Para obter mais informações, consulte Listagem de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListUsers na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(ctx context.Context, maxUsers int32)
 ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(ctx, &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
```

```
 }
 return users, err
}
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
```

```
        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

permissionsBoundary.permissionsBoundaryTypeAsString());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
    }
}
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Liste os usuários.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);

  for (const { UserName, CreateDate } of response.Users) {
    console.log(`${UserName} created on: ${CreateDate}`);
  }
  return response;
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun listAllUsers() {
    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para Kotlin.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
$uuid = uniqid();
$service = new IAMService();
```

```
    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para PHP.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera uma compilação de usuários na atual Conta da AWS.

```
Get-IAMUserList
```

Saída:

```
Arn              : arn:aws:iam::123456789012:user/Administrator
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE1
UserName         : Administrator

Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/6/2015 12:54:42 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : L3EWNONDOM3YUEXAMPLE2
UserName         : bab
```

```
    Arn              : arn:aws:iam::123456789012:user/David
    CreateDate       : 12/10/2014 3:39:27 PM
    PasswordLastUsed : 3/19/2015 8:44:04 AM
    Path             : /
    UserId           : Y4FKWQCXTA52QEXAMPLE3
    UserName         : David
```

- Consulte detalhes da API em ListUsers na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera uma compilação de usuários na atual Conta da AWS.

```
Get-IAMUserList
```

Saída:

```
    Arn              : arn:aws:iam::123456789012:user/Administrator
    CreateDate       : 10/16/2014 9:03:09 AM
    PasswordLastUsed : 3/4/2015 12:12:33 PM
    Path             : /
    UserId           : 7K3GJEANSKZF2EXAMPLE1
    UserName         : Administrator

    Arn              : arn:aws:iam::123456789012:user/Bob
    CreateDate       : 4/6/2015 12:54:42 PM
    PasswordLastUsed : 1/1/0001 12:00:00 AM
    Path             : /
    UserId           : L3EWNONDOM3YUEXAMPLE2
    UserName         : bab

    Arn              : arn:aws:iam::123456789012:user/David
    CreateDate       : 12/10/2014 3:39:27 PM
    PasswordLastUsed : 3/19/2015 8:44:04 AM
    Path             : /
    UserId           : Y4FKWQCXTA52QEXAMPLE3
    UserName         : David
```

- Para obter detalhes sobre a API, consulte ListUsers na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Consulte detalhes da API em ListUsers na Referência da API do AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  # Lists all users in the AWS account
  #
  # @return [Array<Aws::IAM::Types::User>] An array of user objects
  def list_users
    users = []
    @iam_client.list_users.each_page do |page|
      page.users.each do |user|
        users << user
      end
    end
    users
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing users: #{e.message}")
    []
  end
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
```

```
        .await?;
    Ok(response)
}
```

- Consulte detalhes da API em ListUsers na Referência da API AWS SDK for Rust.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    oo_result = lo_iam->listusers( ).
    MESSAGE 'Retrieved user list.' TYPE 'I'.
  CATCH /aws1/cx_iamservicefailureex.
    MESSAGE 'Service failure when listing users.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte ListUsers na Referência de APIs do AWS SDK para SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
```

```
import AWSS3


    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []

        // Use "Paginated" to get all the users.
        // This lets the SDK handle the 'isTruncated' in "ListUsersOutput".
        let input = ListUsersInput()
        let output = client.listUsersPaginated(input: input)

        do {
            for try await page in output {
                guard let users = page.users else {
                    continue
                }
                for user in users {
                    if let id = user.userId, let name = user.userName {
                        userList.append(MyUserRecord(id: id, name: name))
                    }
                }
            }
        }
        catch {
            print("ERROR: listUsers:", dump(error))
            throw error
        }
       return userList
    }
```

- Para obter detalhes da API, consulte ListUsers na referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListVirtualMfaDevices`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ListVirtualMfaDevices`.

CLI

AWS CLI

Listar dispositivos de MFA virtuais

O comando `list-virtual-mfa-devices` a seguir lista os dispositivos de MFA virtuais configurados na conta atual.

```
aws iam list-virtual-mfa-devices
```

Resultado:

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

Para obter mais informações, consulte Habilitar um dispositivo de autenticação multifator (MFA) virtual no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ListVirtualMfaDevices na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo recupera uma compilação dos dispositivos de MFA virtuais atribuídos aos usuários na conta da AWS. A propriedade do **`User`** de cada um é um objeto com detalhes do usuário do IAM ao qual o dispositivo está atribuído.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Saída:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Para ver detalhes da API, consulte ListVirtualMfaDevices na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo recupera uma compilação dos dispositivos de MFA virtuais atribuídos aos usuários na conta da AWS. A propriedade do **`User`** de cada um é um objeto com detalhes do usuário do IAM ao qual o dispositivo está atribuído.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Saída:

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Para obter detalhes sobre a API, consulte ListVirtualMfaDevices na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutGroupPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `PutGroupPolicy`.

CLI

AWS CLI

Como adicionar uma política a um grupo

O comando `put-group-policy`, apresentado a seguir, adiciona uma política ao grupo do IAM denominado `Admins`.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Este comando não produz saída.

A política é definida como um documento JSON no arquivo AdminPolicy.json. (O nome e a extensão do arquivo não têm significado.)

Para obter mais informações, consulte Gerenciamento de políticas do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em PutGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma política em linha denominada **`AppTesterPolicy`** e a incorpora no grupo do IAM **`AppTesters`**. Se já existir uma política em linha com o mesmo nome, ela será substituída. O conteúdo da política JSON vem no arquivo **`apptesterpolicy.json`**. Observe que você deve usar o parâmetro **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Consulte detalhes da API em PutGroupPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma política em linha denominada **`AppTesterPolicy`** e a incorpora no grupo do IAM **`AppTesters`**. Se já existir uma política em linha com o mesmo nome, ela será substituída. O conteúdo da política JSON vem no arquivo **`apptesterpolicy.json`**. Observe que você deve usar o parâmetro **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Para obter detalhes sobre a API, consulte PutGroupPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutRolePermissionsBoundary`** com uma CLI

Os exemplos de código a seguir mostram como usar o `PutRolePermissionsBoundary`.

CLI

AWS CLI

Exemplo 1: para aplicar um limite de permissões a um perfil do IAM com base em uma política personalizada

O exemplo de `put-role-permissions-boundary` a seguir aplica a política personalizada denominada `intern-boundary` como limite de permissões no perfil do IAM especificado.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
```

```
    --role-name lambda-application-role
```

Este comando não produz saída.

Exemplo 2: para aplicar um limite de permissões a um perfil do IAM com base em uma política gerenciada da AWS

O exemplo de `put-role-permissions-boundary` a seguir aplica a política gerenciada `PowerUserAccess` da AWS como limite de permissões no perfil do IAM especificado.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

Este comando não produz saída.

Para obter mais informações, consulte Modificar um perfil no Guia do usuário do AWS IAM.

- Consulte detalhes da API em PutRolePermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo mostra como definir o limite de permissões de um perfil do IAM. Você pode definir políticas gerenciadas da AWS ou políticas personalizadas como limite de permissões.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Consulte detalhes da API em PutRolePermissionsBoundary na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo mostra como definir o limite de permissões de um perfil do IAM. Você pode definir políticas gerenciadas da AWS ou políticas personalizadas como limite de permissões.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Para obter detalhes sobre a API, consulte PutRolePermissionsBoundary na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutRolePolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `PutRolePolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conceitos básicos de data warehouses de tecnologia sem servidor
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de clusters de data warehouse provisionados
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
```

```
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Consulte detalhes da API em PutRolePolicy na Referência da API AWS SDK para .NET.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
```

```
    request.SetPolicyDocument(policyDocument);

    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em PutRolePolicy na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como anexar uma política de permissões a um perfil do IAM

O comando `put-role-policy`, apresentado a seguir, adiciona uma política de permissões ao perfil denominada `Test-Role`.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Este comando não produz saída.

A política é definida como um documento JSON no arquivo AdminPolicy.json. (O nome e a extensão do arquivo não têm significado.)

Para anexar uma política de confiança a um perfil, use o comando `update-assume-role-policy`.

Para obter mais informações, consulte Modificar um perfil no Guia do usuário do AWS IAM.

- Consulte detalhes da API em PutRolePolicy na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::amzn-s3-demo-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
```

```
});

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Consulte detalhes da API em PutRolePolicy na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma política em linha denominada **`FedTesterRolePolicy`** e a incorpora no perfil do IAM **`FedTesterRole`**. Se já existir uma política em linha com o mesmo nome, ela será substituída. O conteúdo da política JSON vem do arquivo **`FedTesterPolicy.json`**. Observe que você deve usar o parâmetro **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Consulte detalhes da API em PutRolePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma política em linha denominada **`FedTesterRolePolicy`** e a incorpora no perfil do IAM **`FedTesterRole`**. Se já existir uma política em linha com o mesmo nome, ela será substituída. O conteúdo da política JSON vem do arquivo **`FedTesterPolicy.json`**. Observe que você deve usar o parâmetro **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Para obter detalhes sobre a API, consulte PutRolePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutUserPermissionsBoundary`** com uma CLI

Os exemplos de código a seguir mostram como usar o `PutUserPermissionsBoundary`.

CLI

AWS CLI

Exemplo 1: para aplicar um limite de permissões a um usuário do IAM com base em uma política personalizada

O exemplo de `put-user-permissions-boundary` a seguir aplica uma política personalizada denominada `intern-boundary` como limite de permissões no usuário do IAM especificado.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

Este comando não produz saída.

Exemplo 2: para aplicar um limite de permissões a um usuário do IAM com base em uma política gerenciada da AWS

O exemplo de `put-user-permissions-boundary` a seguir aplica a política gerenciada da AWS denominada `PowerUserAccess` como limite de permissões no usuário do IAM especificado.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

Este comando não produz saída.

Para ter mais informações, consulte Adicionar e remover permissões de identidade do IAM no AWSGuia do usuário do IAM.

- Consulte detalhes da API em PutUserPermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo mostra como definir o limite de permissões do usuário. Você pode definir políticas gerenciadas da AWS ou políticas personalizadas como limite de permissões.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Consulte detalhes da API em PutUserPermissionsBoundary na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo mostra como definir o limite de permissões do usuário. Você pode definir políticas gerenciadas da AWS ou políticas personalizadas como limite de permissões.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Para obter detalhes sobre a API, consulte PutUserPermissionsBoundary na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutUserPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `PutUserPolicy`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Conheça os conceitos básicos

CLI

AWS CLI

Como anexar uma política a um usuário do IAM

O comando `put-user-policy`, apresentado a seguir, anexa uma política ao usuário do IAM denominado `Bob`.

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Este comando não produz saída.

A política é definida como um documento JSON no arquivo AdminPolicy.json. (O nome e a extensão do arquivo não têm significado.)

Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em PutUserPolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import (
 "context"
 "encoding/json"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/iam"
 "github.com/aws/aws-sdk-go-v2/service/iam/types"
 "github.com/aws/smithy-go"
)

// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(ctx context.Context, userName string,
 policyName string, actions []string,
 roleArn string) error {
 policyDoc := PolicyDocument{
  Version: "2012-10-17",
```

```
  Statement: []PolicyStatement{{
   Effect:   "Allow",
   Action:   actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(ctx, &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Consulte detalhes da API em PutUserPolicy na Referência da API AWS SDK para Go.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo cria uma política em linha denominada **EC2AccessPolicy** e a incorpora no usuário do IAM **Bob**. Se já existir uma política em linha com o mesmo nome, ela será substituída. O conteúdo da política JSON vem do arquivo **EC2AccessPolicy.json**. Observe que você deve usar o parâmetro **-Raw** para processar com êxito o conteúdo do arquivo JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Consulte detalhes da API em PutUserPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo cria uma política em linha denominada **EC2AccessPolicy** e a incorpora no usuário do IAM **Bob**. Se já existir uma política em linha com o mesmo nome, ela será substituída. O conteúdo da política JSON vem do arquivo **EC2AccessPolicy.json**. Observe que você deve usar o parâmetro **-Raw** para processar com êxito o conteúdo do arquivo JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Para obter detalhes sobre a API, consulte PutUserPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# Creates an inline policy for a specified user.
# @param username [String] The name of the IAM user.
# @param policy_name [String] The name of the policy to create.
# @param policy_document [String] The JSON policy document.
# @return [Boolean]
def create_user_policy(username, policy_name, policy_document)
  @iam_client.put_user_policy({
                                user_name: username,
                                policy_name: policy_name,
                                policy_document: policy_document
                              })
  @logger.info("Policy #{policy_name} created for user #{username}.")
  true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    false
  end
```

- Consulte detalhes da API em PutUserPolicy na Referência da API AWS SDK para Ruby.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSIAM
import AWSS3


    func putUserPolicy(policyDocument: String, policyName: String, user:
 IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            print("ERROR: putUserPolicy:", dump(error))
            throw error
        }
    }
```

- Para obter detalhes sobre a API, consulte PutUserPolicy na Referência da API do AWS SDK para Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`RemoveClientIdFromOpenIdConnectProvider`** com uma CLI

Os exemplos de código a seguir mostram como usar o `RemoveClientIdFromOpenIdConnectProvider`.

CLI

AWS CLI

Para remover o ID do cliente especificado da lista de IDs de clientes registrados do provedor OpenID Connect do IAM

Este exemplo remove o ID do cliente `My-TestApp-3` da lista de IDs de cliente associados ao provedor OIDC do IAM cujo ARN é `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`.

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Este comando não produz saída.

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em RemoveClientIdFromOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo remove o ID do cliente **My-TestApp-3** da lista de IDs de cliente associados ao provedor OIDC do IAM cujo ARN é **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**.

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Consulte detalhes da API em RemoveClientIdFromOpenIdConnectProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo remove o ID do cliente **My-TestApp-3** da lista de IDs de cliente associados ao provedor OIDC do IAM cujo ARN é **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**.

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Para obter detalhes sobre a API, consulte RemoveClientIdFromOpenIdConnectProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **RemoveRoleFromInstanceProfile** com uma CLI

Os exemplos de código a seguir mostram como usar o `RemoveRoleFromInstanceProfile`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Conceitos básicos de streaming gerenciado
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

CLI

AWS CLI

Para remover um cargo de um perfil de instância

O comando `remove-role-from-instance-profile` a seguir remove o perfil denominado `Test-Role` do perfil de instância denominado `ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

Para obter mais informações, consulte Usar perfis de instância no Guia do usuário do AWS IAM.

- Consulte detalhes da API em RemoveRoleFromInstanceProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo exclui o perfil denominado **`MyNewRole`** do perfil de instância do EC2 denominado **`MyNewRole`**. Um perfil de instância criado no console do IAM sempre tem o mesmo nome do perfil, como neste exemplo. Se você os criar na API ou na CLI, eles poderão ter nomes diferentes.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Consulte detalhes da API em RemoveRoleFromInstanceProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo exclui o perfil denominado **`MyNewRole`** do perfil de instância do EC2 denominado **`MyNewRole`**. Um perfil de instância criado no console do IAM sempre tem o mesmo nome do perfil, como neste exemplo. Se você os criar na API ou na CLI, eles poderão ter nomes diferentes.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Para obter detalhes sobre a API, consulte RemoveRoleFromInstanceProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`RemoveUserFromGroup`** com uma CLI

Os exemplos de código a seguir mostram como usar o `RemoveUserFromGroup`.

CLI

AWS CLI

Como remover um usuário de um grupo do IAM

O comando `remove-user-from-group`, apresentado a seguir, remove o usuário denominado `Bob` do grupo do IAM denominado `Admins`.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Este comando não produz saída.

Para obter mais informações, consulte Adicionar e remover usuários de um grupo de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em RemoveUserFromGroup na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo remove o usuário do IAM **`Bob`** do grupo **`Testers`**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Exemplo 2: este exemplo encontra todos os grupos dos quais a usuária do IAM **Theresa** é membro e, em seguida, remove a **Theresa** desses grupos.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Exemplo 3: este exemplo mostra uma forma alternativa de remover o usuário do IAM **Bob** do grupo **Testers**.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Consulte detalhes da API em RemoveUserFromGroup na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo remove o usuário do IAM **Bob** do grupo **Testers**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Exemplo 2: este exemplo encontra todos os grupos dos quais a usuária do IAM **Theresa** é membro e, em seguida, remove a **Theresa** desses grupos.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Exemplo 3: este exemplo mostra uma forma alternativa de remover o usuário do IAM **Bob** do grupo **Testers**.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Para obter detalhes sobre a API, consulte RemoveUserFromGroup na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ResyncMfaDevice`** com uma CLI

Os exemplos de código a seguir mostram como usar o `ResyncMfaDevice`.

CLI

AWS CLI

Para sincronizar um dispositivo de MFA

O exemplo de `resync-mfa-device` a seguir sincroniza o dispositivo de MFA associado ao usuário do IAM Bob e cujo ARN é `arn:aws:iam::123456789012:mfa/BobsMFADevice` com um programa autenticador que forneceu os dois códigos de autenticação.

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

Este comando não produz saída.

Para obter mais informações, consulte Uso de autenticação multifator (MFA) na AWS no Guia do usuário do AWS IAM.

- Consulte detalhes da API em ResyncMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo sincroniza o dispositivo de MFA associado ao usuário do IAM **`Bob`** e cujo ARN é **`arn:aws:iam::123456789012:mfa/bob`** com um programa autenticador que forneceu os dois códigos de autenticação.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Exemplo 2: este exemplo sincroniza o dispositivo de MFA do IAM associado à usuária do IAM **`Theresa`** com um dispositivo físico que tem o número de série **`ABCD12345678`** e que forneceu os dois códigos de autenticação.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Consulte detalhes da API em ResyncMfaDevice na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo sincroniza o dispositivo de MFA associado ao usuário do IAM **`Bob`** e cujo ARN é **`arn:aws:iam::123456789012:mfa/bob`** com um programa autenticador que forneceu os dois códigos de autenticação.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Exemplo 2: este exemplo sincroniza o dispositivo de MFA do IAM associado à usuária do IAM **`Theresa`** com um dispositivo físico que tem o número de série **`ABCD12345678`** e que forneceu os dois códigos de autenticação.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Para obter detalhes sobre a API, consulte ResyncMfaDevice na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`SetDefaultPolicyVersion`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `SetDefaultPolicyVersion`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Políticas gerenciadas

- Reverter uma versão de política

CLI

AWS CLI

Para definir a versão especificada da política especificada como a versão da política padrão.

Este exemplo define a versão v2 da política cujo ARN é `arn:aws:iam::123456789012:policy/MyPolicy` como versão ativa padrão.

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Para obter mais informações, consulte Políticas e permissões no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em SetDefaultPolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo define a versão **v2** da política cujo ARN é **arn:aws:iam::123456789012:policy/MyPolicy** como versão ativa padrão.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Consulte detalhes da API em SetDefaultPolicyVersion na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo define a versão **v2** da política cujo ARN é **arn:aws:iam::123456789012:policy/MyPolicy** como versão ativa padrão.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Para obter detalhes sobre a API, consulte SetDefaultPolicyVersion na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
TRY.
    lo_iam->setdefaultpolicyversion(
      iv_policyarn = iv_policy_arn
      iv_versionid = iv_version_id ).
    MESSAGE 'Default policy version set successfully.' TYPE 'I'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'Policy or version does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'Invalid input provided.' TYPE 'E'.
  CATCH /aws1/cx_iamlimitexceededex.
    MESSAGE 'Limit exceeded.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte SetDefaultPolicyVersion na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`TagRole`** com uma CLI

Os exemplos de código a seguir mostram como usar o `TagRole`.

CLI

AWS CLI

Adicionar uma tag a um perfil

O comando `tag-role` a seguir adiciona uma tag com o nome do Departamento ao perfil especificado.

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Este comando não produz saída.

Para obter mais informações, consulte Recursos de tags do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em TagRole na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo adiciona uma tag ao perfil no Identity Management Service

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Consulte detalhes da API em TagRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo adiciona uma tag ao perfil no Identity Management Service

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Para obter detalhes sobre a API, consulte TagRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`TagUser`** com uma CLI

Os exemplos de código a seguir mostram como usar o `TagUser`.

CLI

AWS CLI

Adicionar uma tag a um usuário

O comando `tag-user` a seguir adiciona uma tag com o Departamento associado ao usuário especificado.

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Este comando não produz saída.

Para obter mais informações, consulte Recursos de tags do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em TagUser na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo adiciona uma tag ao usuário no Identity Management Service

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Consulte detalhes da API em TagUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo adiciona uma tag ao usuário no Identity Management Service

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Para obter detalhes sobre a API, consulte TagUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UntagRole`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UntagRole`.

CLI

AWS CLI

Remover uma tag de um perfil

O comando `untag-role` a seguir remove qualquer tag com o nome de chave 'Department' do perfil especificado.

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

Este comando não produz saída.

Para obter mais informações, consulte Recursos de tags do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UntagRole na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo remove a tag do perfil denominado “MyRoleName” com a chave de tag como "abac". Para remover várias tags, forneça uma lista de chaves de tag separadas por vírgulas.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Consulte detalhes da API em UntagRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo remove a tag do perfil denominado “MyRoleName” com a chave de tag como "abac". Para remover várias tags, forneça uma lista de chaves de tag separadas por vírgulas.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Para obter detalhes sobre a API, consulte UntagRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UntagUser`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UntagUser`.

CLI

AWS CLI

Remover uma tag de um usuário

O comando `untag-user` a seguir remove qualquer tag com o nome de chave 'Department' do usuário especificado.

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

Este comando não produz saída.

Para obter mais informações, consulte Recursos de tags do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UntagUser na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo remove a tag do usuário denominado “joe” com a chave de tag como “abac” e “xyzw”. Para remover várias tags, forneça uma lista de chaves de tag separadas por vírgulas.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Consulte detalhes da API em UntagUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo remove a tag do usuário denominado “joe” com a chave de tag como “abac” e “xyzw". Para remover várias tags, forneça uma lista de chaves de tag separadas por vírgulas.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Para obter detalhes sobre a API, consulte UntagUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateAccessKey`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `UpdateAccessKey`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Gerenciar chaves de acesso

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
##############################################################################
# function iam_update_access_key
#
# This function can activate or deactivate an IAM access key for the specified
 IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to update.
#       -a            -- Activate the selected access key.
#       -d            -- Deactivate the selected access key.
#
# Example:
#       # To deactivate the selected access key for IAM user Bob
#       iam_update_access_key -u Bob -k AKIAIOSFODNN7EXAMPLE -d
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_update_access_key() {
  local user_name access_key status response
  local option OPTARG # Required to use getopts command in a function.
  local activate_flag=false deactivate_flag=false

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_update_access_key"
    echo "Updates the status of an AWS Identity and Access Management (IAM)
 access key for the specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to update."
```

```
    echo "  -a              Activate the access key."
    echo "  -d              Deactivate the access key."
    echo ""
  }

  # Retrieve the calling parameters.
    while getopts "u:k:adh" option; do
      case "${option}" in
        u) user_name="${OPTARG}" ;;
        k) access_key="${OPTARG}" ;;
        a) activate_flag=true ;;
        d) deactivate_flag=true ;;
        h)
          usage
          return 0
          ;;
        \?)
          echo "Invalid parameter"
          usage
          return 1
          ;;
      esac
    done
    export OPTIND=1

   # Validate input parameters
    if [[ -z "$user_name" ]]; then
      errecho "ERROR: You must provide a username with the -u parameter."
      usage
      return 1
    fi

    if [[ -z "$access_key" ]]; then
      errecho "ERROR: You must provide an access key with the -k parameter."
      usage
      return 1
    fi

    # Ensure that only -a or -d is specified
    if [[ "$activate_flag" == true && "$deactivate_flag" == true ]]; then
      errecho "ERROR: You cannot specify both -a (activate) and -d (deactivate)
at the same time."
      usage
      return 1
```

```
    fi

    # If neither -a nor -d is provided, return an error
    if [[ "$activate_flag" == false && "$deactivate_flag" == false ]]; then
      errecho "ERROR: You must specify either -a (activate) or -d (deactivate)."
      usage
      return 1
    fi

    # Determine the status based on the flag
    if [[ "$activate_flag" == true ]]; then
      status="Active"
    elif [[ "$deactivate_flag" == true ]]; then
      status="Inactive"
    fi

    iecho "Parameters:\n"
    iecho "    Username:   $user_name"
    iecho "    Access key: $access_key"
    iecho "    New status: $status"
    iecho ""

    # Update the access key status
    response=$(aws iam update-access-key \
      --user-name "$user_name" \
      --access-key-id "$access_key" \
      --status "$status" 2>&1)

    local error_code=${?}

    if [[ $error_code -ne 0 ]]; then
      aws_cli_error_log $error_code
      errecho "ERROR: AWS reports update-access-key operation failed.\n$response"
      return 1
    fi

    iecho "update-access-key response: $response"
    iecho

    return 0
}
```

- Consulte detalhes da API em UpdateAccessKey na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em UpdateAccessKey na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como ativar ou desativar uma chave de acesso para um usuário do IAM

O comando `update-access-key`, apresentado a seguir, desativa a chave de acesso especificada (ID da chave de acesso e chave de acesso secreta) para o usuário do IAM denominado `Bob`.

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Este comando não produz saída.

A desativação da chave significa que ela não pode ser usada para acesso programático à AWS. No entanto, a chave continua disponível e pode ser ativada novamente.

Para obter mais informações, consulte Gerenciamento de chaves de acesso de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateAccessKey na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessId> <status>\s

                Where:
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }
```

```
    public static void updateKey(IamClient iam, String username, String accessId,
 String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
            System.out.printf("Successfully updated the status of access key %s
 to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em UpdateAccessKey na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Atualize a chave de acesso.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em UpdateAccessKey na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em UpdateAccessKey na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo altera para **`Inactive`** o status da chave de acesso **`AKIAIOSFODNN7EXAMPLE`** do usuário do IAM denominado **`Bob`**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Consulte detalhes da API em UpdateAccessKey na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo altera para **`Inactive`** o status da chave de acesso **`AKIAIOSFODNN7EXAMPLE`** do usuário do IAM denominado **`Bob`**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Para obter detalhes sobre a API, consulte UpdateAccessKey na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
```

```
    :param activate: When True, the key is activated. Otherwise, the key is
deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
key_id
        )
        raise
```

- Consulte detalhes da API em UpdateAccessKey, na Referência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
    TRY.
        lo_iam->updateaccesskey(
          iv_accesskeyid = iv_access_key_id
          iv_status = iv_status
          iv_username = iv_user_name ).
        MESSAGE 'Access key updated successfully.' TYPE 'I'.
```

```
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'Access key or user does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte UpdateAccessKey na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateAccountPasswordPolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateAccountPasswordPolicy`.

CLI

AWS CLI

Definir ou alterar a política de senha da conta atual

O comando `update-account-password-policy` a seguir define a política de senha para exigir um mínimo de oito caracteres e um ou mais números na senha.

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

Este comando não produz saída.

As alterações na política de senha de uma conta afetam todas as novas senhas criadas para usuários do IAM na conta. As alterações na política de senha não afetam as senhas existentes.

Para obter mais informações, consulte Definição de uma política de senhas de contas para usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateAccountPasswordPolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza a política de senha da conta com as configurações especificadas. Observe que quaisquer parâmetros que não estejam incluídos no comando não são modificados. Em vez disso, eles são redefinidos para os valores padrão.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Consulte detalhes da API em UpdateAccountPasswordPolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza a política de senha da conta com as configurações especificadas. Observe que quaisquer parâmetros que não estejam incluídos no comando não são modificados. Em vez disso, eles são redefinidos para os valores padrão.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Para obter detalhes sobre a API, consulte UpdateAccountPasswordPolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateAssumeRolePolicy`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateAssumeRolePolicy`.

CLI

AWS CLI

Atualizar a política de confiança de um perfil do IAM

O comando `update-assume-role-policy` a seguir atualiza a política de confiança do perfil denominado `Test-Role`.

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

Este comando não produz saída.

A política de confiança é definida como um documento JSON no arquivo Test-Role-Trust-Policy.json. (O nome e a extensão do arquivo não têm significado.) A política de confiança deve especificar uma entidade principal.

Para atualizar a política de permissões de um perfil, use o comando `put-role-policy`.

Para obter mais informações, consulte Criação de perfis do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateAssumeRolePolicy na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza o perfil do IAM denominado **`ClientRole`** com uma nova política de confiança, cujo conteúdo vem do arquivo **`ClientRolePolicy.json`**. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Consulte detalhes da API em UpdateAssumeRolePolicy na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza o perfil do IAM denominado **`ClientRole`** com uma nova política de confiança, cujo conteúdo vem do arquivo **`ClientRolePolicy.json`**. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Para obter detalhes sobre a API, consulte UpdateAssumeRolePolicy na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateGroup`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateGroup`.

CLI

AWS CLI

Para renomear um grupo do IAM

O comando `update-group` a seguir altera o nome do grupo do IAM `Test` para `Test-1`.

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

Este comando não produz saída.

Para obter mais informações, consulte Renomeação de um grupo de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateGroup na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo renomeia o grupo do IAM **`Testers`** para **`AppTesters`**.

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Exemplo 2: este exemplo altera o caminho do grupo do IAM **`AppTesters`** para **`/Org1/Org2/`**. Isso altera o ARN do grupo para **`arn:aws:iam::123456789012:group/Org1/Org2/AppTesters`**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Consulte detalhes da API em UpdateGroup na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo renomeia o grupo do IAM **`Testers`** para **`AppTesters`**.

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Exemplo 2: este exemplo altera o caminho do grupo do IAM **`AppTesters`** para **`/Org1/Org2/`**. Isso altera o ARN do grupo para **`arn:aws:iam::123456789012:group/Org1/Org2/AppTesters`**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Para obter detalhes sobre a API, consulte UpdateGroup na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateLoginProfile`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateLoginProfile`.

CLI

AWS CLI

Atualizar a senha de um usuário do IAM

O comando `update-login-profile` a seguir cria uma senha para o usuário do IAM chamado Bob.

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

Este comando não produz saída.

Para definir uma política de senha da conta, use o comando `update-account-password-policy`. Se a nova senha violar a política de senha da conta, o comando retornará um erro `PasswordPolicyViolation`.

Se a política de senha da conta permitir, os usuários do IAM poderão alterar suas próprias senhas usando o comando `change-password`.

Armazene a senha em um lugar seguro. Se a senha for perdida, não será possível recuperá-la e você deverá criar uma nova usando o comando `create-login-profile`.

Para obter mais informações, consulte Gerenciamento de senhas de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo define uma nova senha temporária para o usuário **Bob** do IAM e exige que a pessoa altere a senha na próxima vez que fizer login.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Consulte detalhes da API em UpdateLoginProfile na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo define uma nova senha temporária para o usuário **Bob** do IAM e exige que a pessoa altere a senha na próxima vez que fizer login.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Para obter detalhes sobre a API, consulte UpdateLoginProfile na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateOpenIdConnectProviderThumbprint`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateOpenIdConnectProviderThumbprint`.

CLI

AWS CLI

Para substituir a lista existente de impressões digitais do certificado de servidor por uma nova

Este exemplo atualiza a lista de impressões digitais do certificado do provedor OIDC cujo ARN é `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`, a fim de usar uma nova impressão digital.

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

Se a descoberta do seu provedor OIDC e os endpoints do JWKS usarem certificados diferentes, inclua impressões digitais de ambos os endpoints no valor `--thumbprint-list`.

Este comando não produz saída.

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateOpenIdConnectProviderThumbprint na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza a lista de impressões digitais do certificado do provedor OIDC cujo ARN é **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**, a fim de usar uma nova impressão digital. O provedor OIDC compartilha o novo valor quando o certificado associado ao provedor é alterado.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Consulte detalhes da API em UpdateOpenIdConnectProviderThumbprint na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza a lista de impressões digitais do certificado do provedor OIDC cujo ARN é **arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com**, a fim de usar uma nova impressão digital. O provedor OIDC compartilha o novo valor quando o certificado associado ao provedor é alterado.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Para obter detalhes sobre a API, consulte UpdateOpenIdConnectProviderThumbprint na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateRole`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateRole`.

CLI

AWS CLI

Alterar a descrição ou a duração da sessão de um perfil do IAM

O comando `update-role` a seguir altera a descrição do perfil do IAM `production-role` para `Main production role` e define a duração máxima da sessão como 12 horas.

```
aws iam update-role \
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

Este comando não produz saída.

Para obter mais informações, consulte Modificar um perfil no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateRole na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza a descrição do perfil e o valor máximo da duração da sessão (em segundos) para o qual a sessão de um perfil pode ser solicitada.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Consulte detalhes da API em UpdateRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza a descrição do perfil e o valor máximo da duração da sessão (em segundos) para o qual a sessão de um perfil pode ser solicitada.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Para obter detalhes sobre a API, consulte UpdateRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateRoleDescription`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateRoleDescription`.

CLI

AWS CLI

Alterar a descrição de um perfil do IAM

O comando `update-role` a seguir altera a descrição do perfil do IAM `production-role` para `Main production role`.

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

Resultado:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version":"2012-10-17",
            "Statement": [
                {
```

```
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

Para obter mais informações, consulte Modificar um perfil no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateRoleDescription na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza a descrição de um perfil do IAM na sua conta.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Consulte detalhes da API em UpdateRoleDescription na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza a descrição de um perfil do IAM na sua conta.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Para obter detalhes sobre a API, consulte UpdateRoleDescription na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **UpdateSamlProvider** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateSamlProvider`.

CLI

AWS CLI

Atualizar o documento de metadados de um provedor SAML existente

Este exemplo atualiza o provedor SAML no IAM cujo ARN é `arn:aws:iam::123456789012:saml-provider/SAMLADFS` com um novo documento de metadados SAML do arquivo `SAMLMetaData.xml`.

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Resultado:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

Para obter mais informações, consulte Criação de provedores de identidade SAML do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateSamlProvider na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza o provedor SAML no IAM cujo ARN é **arn:aws:iam::123456789012:saml-provider/SAMLADFS** com um novo documento de metadados SAML do arquivo **SAMLMetaData.xml**. Observe que você deve usar o parâmetro switch **-Raw** para processar com êxito o conteúdo do arquivo JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Consulte detalhes da API em UpdateSamlProvider na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza o provedor SAML no IAM cujo ARN é **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`** com um novo documento de metadados SAML do arquivo **`SAMLMetaData.xml`**. Observe que você deve usar o parâmetro switch **`-Raw`** para processar com êxito o conteúdo do arquivo JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Para obter detalhes sobre a API, consulte UpdateSamlProvider na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateServerCertificate`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `UpdateServerCertificate`.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                           const Aws::String &newCertificateName,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
```

```
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- Consulte detalhes da API em UpdateServerCertificate na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como alterar o caminho ou o nome de um certificado de servidor em sua conta da AWS

O comando `update-server-certificate`, apresentado a seguir, altera o nome do certificado de `myServerCertificate` para `myUpdatedServerCertificate`. Além disso, ele altera o caminho para `/cloudfront/` com a finalidade de que ele possa ser acessado

pelo serviço do Amazon CloudFront. Este comando não produz saída. É possível visualizar os resultados da atualização ao executar o comando `list-server-certificates`.

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Este comando não produz saída.

Para obter mais informações, consulte Gerenciar certificados de servidor no IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateServerCertificate na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Atualize um certificado do servidor.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });
```

```
  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em UpdateServerCertificate na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.

- Consulte detalhes da API em UpdateServerCertificate na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo renomeia o certificado denominado **MyServerCertificate** para **MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Exemplo 2: este exemplo move o certificado denominado **MyServerCertificate** para o caminho /Org1/Org2/. Isso altera o ARN do recurso para **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Consulte detalhes da API em UpdateServerCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo renomeia o certificado denominado **MyServerCertificate** para **MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Exemplo 2: este exemplo move o certificado denominado **MyServerCertificate** para o caminho /Org1/Org2/. Isso altera o ARN do recurso para **arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Para obter detalhes sobre a API, consulte UpdateServerCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Listar, atualizar e excluir certificados de servidor.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = 'ServerCertificateManager'
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates
```

```
    if response.server_certificate_metadata_list.empty?
      @logger.info('No server certificates found.')
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Consulte detalhes da API em UpdateServerCertificate na Referência da API AWS SDK para Ruby.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateSigningCertificate`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateSigningCertificate`.

CLI

AWS CLI

Ativar ou desativar um certificado de assinatura de um usuário do IAM

O comando `update-signing-certificate` a seguir desativa o certificado de assinatura especificado do usuário do IAM chamado Bob.

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Para obter o ID de um certificado de assinatura, use o comando `list-signing-certificates`.

Para obter mais informações, consulte Manage signing certificates no Guia do usuário do Amazon EC2.

- Consulte detalhes da API em UpdateSigningCertificate na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo atualiza o certificado associado ao usuário do IAM chamado **Bob** e cujo ID do certificado é **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** para marcá-lo como inativo.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Consulte detalhes da API em UpdateSigningCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo atualiza o certificado associado ao usuário do IAM chamado **Bob** e cujo ID do certificado é **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** para marcá-lo como inativo.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Para obter detalhes sobre a API, consulte UpdateSignnagCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateUser`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `UpdateUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Criar usuários somente leitura e leitura/gravação usando

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
```

```
                            const Aws::String &newUserName,
                            const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como alterar o nome de um usuário do IAM

O comando `update-user`, apresentado a seguir, altera o nome do usuário do IAM de `Bob` para `Robert`.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Este comando não produz saída.

Para obter mais informações, consulte Renomeação de um grupo de usuários do IAM no Guia do usuário do AWS IAM.

- Consulte detalhes da API em UpdateUser na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
```

```
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Atualize o usuário.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK para JavaScript.
- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK para JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?,
) {
```

```
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient.fromEnvironment { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK para Kotlin.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo renomeia o usuário do IAM **Bob** para **Robert**.

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Exemplo 2: este exemplo altera o caminho do usuário do IAM **Bob** para **/Org1/Org2/**, o que efetivamente altera o ARN do usuário para **arn:aws:iam::123456789012:user/Org1/Org2/bob**.

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Consulte detalhes da API em UpdateUser na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo renomeia o usuário do IAM **Bob** para **Robert**.

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Exemplo 2: este exemplo altera o caminho do usuário do IAM **Bob** para **/Org1/Org2/**, o que efetivamente altera o ARN do usuário para **arn:aws:iam::123456789012:user/Org1/Org2/bob**.

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Para obter detalhes sobre a API, consulte UpdateUser na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
  false
end
```

- Consulte detalhes da API em UpdateUser na Referência da API AWS SDK para Ruby.

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
  TRY.
      lo_iam->updateuser(
        iv_username = iv_user_name
        iv_newusername = iv_new_user_name ).
```

```
        MESSAGE 'User updated successfully.' TYPE 'I'.
      CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'User does not exist.' TYPE 'E'.
      CATCH /aws1/cx_iamentityalrdyexex.
        MESSAGE 'New user name already exists.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte UpdateUser na Referência de APIs do AWS SDK para SAP ABAP.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UploadServerCertificate`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `UploadServerCertificate`.

CLI

AWS CLI

Como fazer upload de um certificado de servidor para sua conta da AWS

O comando upload-server-certificate, apresentado a seguir, faz o upload de um certificado de servidor para sua conta da AWS. Neste exemplo, o certificado está no arquivo `public_key_cert_file.pem`, a chave privada associada está no arquivo `my_private_key.pem` e a cadeia de certificados fornecida pela autoridade de certificação (CA) está no arquivo `my_certificate_chain_file.pem`. Quando o upload do arquivo for concluído, ele estará disponível com o nome myServerCertificate. Os parâmetros que começam com `file://` informam ao comando para ler o conteúdo do arquivo e usá-lo como valor do parâmetro em vez do próprio nome do arquivo.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Resultado:

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Para obter mais informações, consulte Creating, Uploading, and Deleting Server Certificates no guia Using IAM.

- Consulte detalhes da API em UploadServerCertificate na Referência de comandos da AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "node:fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "node:path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
```

```
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Consulte detalhes da API em UploadServerCertificate na Referência da API do AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo faz upload de um novo certificado de servidor na conta do IAM. Os arquivos contendo o corpo do certificado, a chave privada e (opcionalmente) a cadeia de certificação devem ser codificados em PEM. Observe que os parâmetros exigem o conteúdo real dos arquivos em vez dos nomes deles. Você deve usar o parâmetro switch **`-Raw`** para processar com êxito o conteúdo do arquivo.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Saída:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Consulte detalhes da API em UploadServerCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo faz upload de um novo certificado de servidor na conta do IAM. Os arquivos contendo o corpo do certificado, a chave privada e (opcionalmente) a cadeia de certificação devem ser codificados em PEM. Observe que os parâmetros exigem o conteúdo real dos arquivos em vez dos nomes deles. Você deve usar o parâmetro switch **`-Raw`** para processar com êxito o conteúdo do arquivo.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Saída:

```
Arn                   : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration            : 1/14/2018 9:52:36 AM
```

```
Path                  : /
ServerCertificateId   : ASCAJIEXAMPLE7J7HQZYW
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Para obter detalhes sobre a API, consulte UploadServerCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UploadSigningCertificate`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UploadSigningCertificate`.

CLI

AWS CLI

Fazer upload de um certificado de assinatura de um usuário do IAM

O comando `upload-signing-certificate` a seguir faz upload de um certificado de assinatura do usuário do IAM chamado `Bob`.

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

Resultado:

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

O certificado está em um arquivo denominado certificate.pem no formato PEM.

Para obter mais informações, consulte Creating and Uploading a User Signing Certificate no guia Uso do IAM.

- Consulte detalhes da API em UploadSigningCertificate na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: este exemplo faz upload de um novo certificado de assinatura X.509 e o associa ao usuário do IAM chamado **Bob**. O arquivo que contém o corpo do certificado é codificado em PEM. O parâmetro **CertificateBody** exige o conteúdo real do arquivo de certificado em vez do nome do arquivo. Você deve usar o parâmetro switch **-Raw** para processar o arquivo com êxito.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Saída:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                   YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                   21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
```

```
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Consulte detalhes da API em UploadSigningCertificate na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: este exemplo faz upload de um novo certificado de assinatura X.509 e o associa ao usuário do IAM chamado **Bob**. O arquivo que contém o corpo do certificado é codificado em PEM. O parâmetro **CertificateBody** exige o conteúdo real do arquivo de certificado em vez do nome do arquivo. Você deve usar o parâmetro switch **-Raw** para processar o arquivo com êxito.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Saída:

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
```

```
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Para obter detalhes sobre a API, consulte UploadSignnagCertificate na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cenários para o IAM usando AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns no IAM com AWS SDKs. Esses cenários mostram como realizar tarefas específicas chamando várias funções dentro do IAM ou combinadas com outros Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, em que você pode encontrar instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudar você a compreender ações de serviço em contexto.

Exemplos

- Criar e gerenciar um serviço resiliente usando um AWS SDK
- Configurar a conectividade do serviço de contêiner
- Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
- Criar uma API REST com integração de proxy de função
- Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Como criar sua primeira função com tecnologia sem servidor
- Conceitos básicos de data warehouses com tecnologia sem servidor usando a CLI
- Conceitos básicos do gerenciamento de configurações
- Conceitos básicos da proteção de dispositivos da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos de clusters de data warehouse provisionados
- Conceitos básicos da orquestração de fluxo de trabalaho
- Gerenciar chaves de acesso do IAM usando um AWS SDK
- Gerenciar políticas do IAM usando um AWS SDK
- Gerenciar perfis do IAM usando um AWS SDK
- Gerenciar a conta do IAM usando um AWS SDK
- Como mover segredos codificados para o armazenamento seguro de segredos
- Permite que o atributo de automação AWS Compute Optimizer aplique as ações recomendadas
- Política para permitir a automação em toda a sua organização
- Política para permitir a automação na sua conta
- Política para conceder acesso total ao Compute Optimizer Automation para uma conta gerencial de uma organização
- Política para conceder acesso total ao Compute Optimizer Automation para contas autônomas da AWS

- Política para conceder acesso somente leitura ao Compute Optimizer Automation para uma conta gerencial de uma organização
- Política para conceder acesso somente leitura ao Compute Optimizer Automation para contas autônomas da AWS
- Política para conceder permissões de perfil vinculado ao serviço para o Compute Optimization Automation
- Reverter uma versão de política do IAM usando um AWS SDK
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Configurar o controle de acesso por atributo para o DynamoDB usando a AWS Command Line Interface v2
- Como configurar o gerenciamento de sistemas
- Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor
- Trabalhar com fluxos e vida útil do DynamoDB usando a AWS Command Line Interface v2
- Trabalhar com a API IAM Policy Builder usando um AWS SDK

Criar e gerenciar um serviço resiliente usando um AWS SDK

Os exemplos de código a seguir mostram como criar um serviço da Web com carga balanceada que retorna recomendações de livros, filmes e músicas. O exemplo mostra como o serviço responde a falhas e como é possível reestruturá-lo para gerar mais resiliência em caso de falhas.

- Use um grupo do Amazon EC2 Auto Scaling para criar instâncias do Amazon Elastic Compute Cloud (Amazon EC2) com base em um modelo de execução e para manter o número de instâncias em um intervalo especificado.
- Gerencie e distribua solicitações HTTP com o Elastic Load Balancing.
- Monitore a integridade das instâncias em um grupo do Auto Scaling e encaminhe solicitações somente para instâncias íntegras.
- Execute um servidor Web Python em cada instância do EC2 para lidar com solicitações HTTP. O servidor Web responde com recomendações e verificações de integridade.
- Simule um serviço de recomendação com uma tabela do Amazon DynamoDB.
- Controle a resposta do servidor Web às solicitações e verificações de integridade atualizando os parâmetros do AWS Systems Manager.

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Execute o cenário interativo em um prompt de comando.

```
    static async Task Main(string[] args)
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
```

```
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
```

```
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
```

```
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
```

```
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
```

```
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
_autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
_autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
ipString);
            var sshPortIsOpen =
_autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
your default VPC must\n"
                    + "allows access from this computer. You can either add it
automatically from this\n"
                    + "example or add it yourself using the AWS Management
Console.\n");
```

```
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound traffic from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
```

```
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
```

```
                        "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
       if (interactive)
           await DemoActionChoices();

       Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
       Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

       await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

       Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
       Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
       if (interactive)
           await DemoActionChoices();

       Console.WriteLine("Let's reinstate the recommendation service.\n");
       await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
       Console.WriteLine(
           "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
           "access the DynamoDB recommendation table.\n"
       );
       await _autoScalerWrapper.CreateInstanceProfileWithName(
           _autoScalerWrapper.BadCredsPolicyName,
           _autoScalerWrapper.BadCredsRoleName,
           _autoScalerWrapper.BadCredsProfileName,
           ssmOnlyPolicy,
           new List<string> { "AmazonSSMManagedInstanceCore" }
       );
       var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
       var badInstanceId = instances.First();
       var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
       Console.WriteLine(
```

```
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");
```

```
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
```

```
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
_recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
            );
        }
```

```
        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Crie uma classe que envolva ações do Auto Scaling e do Amazon EC2.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;
    private readonly ILogger<AutoScalerWrapper> _logger;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
```

```
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration,
        ILogger<AutoScalerWrapper> logger)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;
        _logger = logger;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
```

```
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
                                         "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
                                    "}]" +
                                "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
```

```
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
```

```
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
```

```
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
```

```
   /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
   /// <returns>The template object.</returns>
   public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
   {
       try
       {
           await CreateKeyPair(_keyPairName);
           await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName,
               _instanceProfileName, instancePolicyPath);

           var startServerText = await File.ReadAllTextAsync(startupScriptPath);
           var plainTextBytes =
System.Text.Encoding.UTF8.GetBytes(startServerText);

           var amiLatest = await _amazonSsm.GetParameterAsync(
               new GetParameterRequest() { Name = _amiParam });
           var amiId = amiLatest.Parameter.Value;
           var launchTemplateResponse = await
_amazonEc2.CreateLaunchTemplateAsync(
               new CreateLaunchTemplateRequest()
               {
                   LaunchTemplateName = _launchTemplateName,
                   LaunchTemplateData = new RequestLaunchTemplateData()
                   {
                       InstanceType = _instanceType,
                       ImageId = amiId,
                       IamInstanceProfile =
                           new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                           {
                               Name = _instanceProfileName
                           },
                       KeyName = _keyPairName,
                       UserData = System.Convert.ToBase64String(plainTextBytes)
                   }
               });
           return launchTemplateResponse.LaunchTemplate;
       }
       catch (AmazonEC2Exception ec2Exception)
       {
```

```
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.AlreadyExistsException")
            {
                _logger.LogError($"Could not create the template, the name
{_launchTemplateName} already exists. " +
                                 $"Please try again with a unique name.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while creating the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        try
        {
            var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
                new DescribeAvailabilityZonesRequest());
            return zoneResponse.AvailabilityZones.Select(z =>
z.ZoneName).ToList();
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            _logger.LogError($"An Amazon EC2 error occurred while listing
availability zones.: {ec2Exception.Message}");
            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while listing availability
zones.: {ex.Message}");
            throw;
        }
```

```
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
 Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
 size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
 exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
```

```
    public async Task<Vpc> GetDefaultVpc()
    {
        try
        {
            var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
                new DescribeVpcsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("is-default", new List<string>() { "true" })
                    }
                });
            return vpcResponse.Vpcs[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "UnauthorizedOperation")
            {
                _logger.LogError(ec2Exception, $"You do not have the necessary
permissions to describe VPCs.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the vpcs.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        try
        {
            var subnets = new List<Subnet>();
```

```
            var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
                new DescribeSubnetsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("vpc-id", new List<string>() { vpcId }),
                        new("availability-zone", availabilityZones),
                        new("default-for-az", new List<string>() { "true" })
                    }
                });

            // Get the entire list using the paginator.
            await foreach (var subnet in subnetPaginator.Subnets)
            {
                subnets.Add(subnet);
            }

            return subnets;
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidVpcID.NotFound")
            {
                _logger.LogError(ec2Exception, $"The specified VPC ID {vpcId}
does not exist.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while describing the
subnets.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
```

```
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode ==
"InvalidLaunchTemplateName.NotFoundException")
            {
                _logger.LogError(
                    $"Could not delete the template, the name
{_launchTemplateName} was not found.");
            }

            throw;
        }
        catch (Exception ex)
        {
            _logger.LogError($"An error occurred while deleting the template.:
{ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
```

```
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
```

```
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        try
        {
            var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
                new DescribeIamInstanceProfileAssociationsRequest()
                {
                    Filters = new List<Amazon.EC2.Model.Filter>()
                    {
                        new("instance-id", new List<string>() { instanceId })
                    },
                });
            return response.IamInstanceProfileAssociations[0];
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
        }
        catch (Exception ex)
        {
```

```
            _logger.LogError(ex, $"An error occurred while creating the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        try
        {
            await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
            // Allow time before resetting.
            Thread.Sleep(25000);

            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(25000);
            var instanceReady = false;
            var retries = 5;
            while (retries-- > 0 && !instanceReady)
            {
                var instancesPaginator =
```

```
                    _amazonSsm.Paginators.DescribeInstanceInformation(
                        new DescribeInstanceInformationRequest());
                // Get the entire list using the paginator.
                await foreach (var instance in
instancesPaginator.InstanceInformationList)
                {
                    instanceReady = instance.InstanceId == instanceId;
                    if (instanceReady)
                    {
                        break;
                    }
                }
            }
            Console.WriteLine("Waiting for instance to be running.");
            await WaitForInstanceState(instanceId, InstanceStateName.Running);
            Console.WriteLine("Instance ready.");
            Console.WriteLine($"Sending restart command to instance
{instanceId}");
            await _amazonSsm.SendCommandAsync(
                new SendCommandRequest()
                {
                    InstanceIds = new List<string>() { instanceId },
                    DocumentName = "AWS-RunShellScript",
                    Parameters = new Dictionary<string, List<string>>()
                    {
                        {
                            "commands",
                            new List<string>() { "cd / && sudo python3 server.py
80" }
                        }
                    }
                });
            Console.WriteLine($"Restarted the web server on instance
{instanceId}");
        }
        catch (AmazonEC2Exception ec2Exception)
        {
            if (ec2Exception.ErrorCode == "InvalidInstanceID.NotFound")
            {
                _logger.LogError(ec2Exception, $"Instance {instanceId} not
found");
            }

            throw;
```

```
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, $"An error occurred while replacing the
template.: {ex.Message}");
            throw;
        }
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
```

```
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
```

```
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
```

```
   /// This can be done by allowing ingress from this computer's IP address.
   /// In some situations, such as connecting from a corporate network, you must
instead specify
   /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
   /// If you do, be sure to remove public access when you're done.
   /// </summary>
   /// <param name="vpc">The group to check.</param>
   /// <param name="port">The port to verify.</param>
   /// <param name="ipAddress">This computer's IP address.</param>
   /// <returns>True if the ip address is allowed on the group.</returns>
   public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
   {
       var portIsOpen = false;
       foreach (var ipPermission in group.IpPermissions)
       {
           if (ipPermission.FromPort == port)
           {
               foreach (var ipRange in ipPermission.Ipv4Ranges)
               {
                   var cidr = ipRange.CidrIp;
                   if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                   {
                       portIsOpen = true;
                   }
               }

               if (ipPermission.PrefixListIds.Any())
               {
                   portIsOpen = true;
               }

               if (!portIsOpen)
               {
                   Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                     "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
               }
               else
               {
                   break;
               }
```

```
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
 the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
```

```
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
 InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is in the specified state.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);

            // Check for the desired state.
            var response = await _amazonEc2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }
}
```

Crie uma classe que envolva ações do Elastic Load Balancing.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
 client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
 loadBalancerName)
    {
```

```
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
```

```
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
 groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
 how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
```

```
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
    /// Create an Elastic Load Balancing load balancer that uses the specified
subnets
    /// and forwards requests to the specified target group.
    /// </summary>
    /// <param name="name">The name for the new load balancer.</param>
    /// <param name="subnetIds">Subnets for the load balancer.</param>
    /// <param name="targetGroup">Target group for forwarded requests.</param>
    /// <returns>The new LoadBalancer object.</returns>
    public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
List<string> subnetIds, TargetGroup targetGroup)
    {
        var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
            new CreateLoadBalancerRequest()
            {
                Name = name,
                Subnets = subnetIds
            });
        var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

        // Wait for load balancer to be available.
        var loadBalancerReady = false;
        while (!loadBalancerReady)
        {
            try
            {
                var describeResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                        new DescribeLoadBalancersRequest()
                        {
                            Names = new List<string>() { name }
                        });
```

```
                var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

                loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
            }
            catch (LoadBalancerNotFoundException)
            {
                loadBalancerReady = false;
            }
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
```

```
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
 describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
```

```
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
targetArn });
                Console.WriteLine($"Deleted load balancing target group
{groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
```

```
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Crie uma classe que use o DynamoDB para simular um serviço de recomendação.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
 configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
```

```
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
```

```
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
 _amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
```

```
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Crie uma classe que envolva as ações do Systems Manager.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
```

```
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
```

```
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para .NET.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs

- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Execute o cenário interativo em um prompt de comando.

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";
```

```
    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
        System.out.println("Welcome to the demonstration of How to Build and
 Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
 resources.");
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
 service.
                To keep things tidy and to avoid unwanted charges on your
 account, we can clean up all AWS resources
                that were created for this demo.
```

```
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
```

```
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
```

```
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
```

```
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
```

```
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
```

```
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
```

```
                        While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                        """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
```

```
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
```

```
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
```

```
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());
```

```
                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
 balancer targets:\n");
                        List<TargetHealthDescription> health =
 loadBalancer.checkTargetHealth(targetGroupName);
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
 target.target().id(),
                                    target.target().port(),
 target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
 health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Crie uma classe que envolva ações do Auto Scaling e do Amazon EC2.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
```

```
   }

   /**
    * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
    * terminated, it can no longer be accessed.
    */
   public void terminateInstance(String instanceId) {
       TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
               .builder()
               .instanceId(instanceId)
               .shouldDecrementDesiredCapacity(false)
               .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
       System.out.format("Terminated instance %s.", instanceId);
   }

   /**
    * Replaces the profile associated with a running instance. After the profile
is
    * replaced, the instance is rebooted to ensure that it uses the new profile.
    * When
    * the instance is ready, Systems Manager is used to restart the Python web
    * server.
    */
   public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
           throws InterruptedException {
       // Create an IAM instance profile specification.
       software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
               .builder()
               .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                             // name.
               .build();

       // Replace the IAM instance profile association for the EC2 instance.
       ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
```

```
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
```

```
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();
```

```
            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");
```

```
        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
     * this
     * computer. This can be done by allowing ingress from this computer's IP
     * address. In some situations, such as connecting from a corporate network,
you
     * must instead specify a prefix list ID. You can also temporarily open the
port
     * to
     * any IP address while running this example. If you do, be sure to remove
     * public
     * access when you're done.
     *
     */
    public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
        boolean portIsOpen = false;
        GroupInfo groupInfo = new GroupInfo();
        try {
            Filter filter = Filter.builder()
                    .name("group-name")
```

```
                    .values("default")
                    .build();

            Filter filter1 = Filter.builder()
                    .name("vpc-id")
                    .values(VPC)
                    .build();

            DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                    .filters(filter, filter1)
                    .build();

            DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                    .describeSecurityGroups(securityGroupsRequest);
            String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
            groupInfo.setGroupName(securityGroup);

            for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
```

```
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
     */
    public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
        try {
            AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                    .autoScalingGroupName(asGroupName)
                    .targetGroupARNs(targetGroupARN)
                    .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
            System.out.println("Attached load balancer to " + asGroupName);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
```

```
   }

   // Creates an EC2 Auto Scaling group with the specified size.
   public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

       // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());

       String availabilityZones = String.join(",", availabilityZoneNames);
       LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
               .launchTemplateName(templateName)
               .version("$Default")
               .build();

       String[] zones = availabilityZones.split(",");
       CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
               .launchTemplate(specification)
               .availabilityZones(zones)
               .maxSize(groupSize)
               .minSize(groupSize)
               .autoScalingGroupName(autoScalingGroupName)
               .build();

       try {
           getAutoScalingClient().createAutoScalingGroup(groupRequest);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
```

```
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();
```

```
        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
 InstanceProfile) {
        ListPoliciesRequest listPoliciesRequest =
 ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
 getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
 the policy.

 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();
```

```
                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Crie uma classe que envolva ações do Elastic Load Balancing.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
 targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
 DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();

        DescribeTargetGroupsResponse tgResponse =
 getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
 DescribeTargetHealthRequest.builder()

 .targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
 getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }
```

```
   // Deletes a load balancer.
   public void deleteLoadBalancer(String lbName) {
       try {
           // Use a waiter to delete the Load Balancer.
           DescribeLoadBalancersResponse res = getLoadBalancerClient()
                   .describeLoadBalancers(describe -> describe.names(lbName));
           ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
           DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                   .build();

           getLoadBalancerClient().deleteLoadBalancer(
                   builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
           WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                   .waitUntilLoadBalancersDeleted(request);
           waiterResponse.matched().response().ifPresent(System.out::println);

       } catch (ElasticLoadBalancingV2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
       }
       System.out.println(lbName + " was deleted.");
   }

   // Deletes the target group.
   public void deleteTargetGroup(String targetGroupName) {
       try {
           DescribeTargetGroupsResponse res = getLoadBalancerClient()
                   .describeTargetGroups(describe ->
describe.names(targetGroupName));
           getLoadBalancerClient()
                   .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
       } catch (ElasticLoadBalancingV2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
       }
       System.out.println(targetGroupName + " was deleted.");
   }
```

```
   // Verify this computer can successfully send a GET request to the load
balancer
   // endpoint.
   public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
       boolean success = false;
       int retries = 3;
       CloseableHttpClient httpClient = HttpClients.createDefault();

       // Create an HTTP GET request to the ELB.
       HttpGet httpGet = new HttpGet("http://" + elbDnsName);
       try {
           while ((!success) && (retries > 0)) {
               // Execute the request and get the response.
               HttpResponse response = httpClient.execute(httpGet);
               int statusCode = response.getStatusLine().getStatusCode();
               System.out.println("HTTP Status Code: " + statusCode);
               if (statusCode == 200) {
                   success = true;
               } else {
                   retries--;
                   System.out.println("Got connection error from load balancer
endpoint, retrying...");
                   TimeUnit.SECONDS.sleep(15);
               }
           }

       } catch (org.apache.http.conn.HttpHostConnectException e) {
           System.out.println(e.getMessage());
       }

       System.out.println("Status.." + success);
       return success;
   }

   /*
    * Creates an Elastic Load Balancing target group. The target group specifies
    * how
    * the load balancer forward requests to instances in the group and how
instance
    * health is checked.
    */
   public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
```

```
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
```

```
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
CreateListenerRequest.builder()

.loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;
```

```
        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Crie uma classe que use o DynamoDB para simular um serviço de recomendação.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
 DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
 e.getMessage());
        }
        return false;
```

```
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
such
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
                                    .attributeName("MediaType")
                                    .attributeType(ScalarAttributeType.S)
                                    .build(),
                            AttributeDefinition.builder()
                                    .attributeName("ItemId")
                                    .attributeType(ScalarAttributeType.N)
                                    .build())
                    .keySchema(
                            KeySchemaElement.builder()
                                    .attributeName("MediaType")
                                    .keyType(KeyType.HASH)
                                    .build(),
                            KeySchemaElement.builder()
                                    .attributeName("ItemId")
                                    .keyType(KeyType.RANGE)
                                    .build())
                    .provisionedThroughput(
                            ProvisionedThroughput.builder()
                                    .readCapacityUnits(5L)
                                    .writeCapacityUnits(5L)
                                    .build())
                    .build();
```

```
            getDynamoDbClient().createTable(createTableRequest);
            System.out.println("Creating table " + tableName + "...");

            // Wait until the Amazon DynamoDB table is created.
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
```

```
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Crie uma classe que envolva as ações do Systems Manager.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
```

```
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Execute o cenário interativo em um prompt de comando.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
```

```
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Resilient Workflow",
    synopsis:
      "node index.js --scenario <deploy | demo | destroy> [-h|--help] [-y|--yes]
 [-v|--verbose]",
    description: "Deploy and interact with scalable EC2 instances.",
  });
}
```

Criar etapas para implantar todos os recursos.

```
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
```

```
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { saveState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
```

```
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
```

```
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
```

```
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
```

```
      ),
    }),
  );
}),
new ScenarioOutput(
  "createdInstanceRole",
  MESSAGES.createdInstanceRole.replace(
    "${INSTANCE_ROLE_NAME}",
    NAMES.instanceRoleName,
  ),
),
new ScenarioOutput(
  "attachingPolicyToRole",
  MESSAGES.attachingPolicyToRole
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
),
new ScenarioAction("attachPolicyToRole", async (state) => {
  const client = new IAMClient({});
  await client.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.instanceRoleName,
      PolicyArn: state.instancePolicyArn,
    }),
  );
}),
new ScenarioOutput(
  "attachedPolicyToRole",
  MESSAGES.attachedPolicyToRole
    .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
    .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
),
new ScenarioOutput(
  "creatingInstanceProfile",
  MESSAGES.creatingInstanceProfile.replace(
    "${INSTANCE_PROFILE_NAME}",
    NAMES.instanceProfileName,
  ),
),
new ScenarioAction("createInstanceProfile", async (state) => {
  const client = new IAMClient({});
  const {
    InstanceProfile: { Arn },
  } = await client.send(
```

```
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
```

```
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
```

```
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    const client = new EC2Client({});
    const { Subnets } = await client.send(
```

```
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
```

```
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
```

```
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
```

```
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      }
      return MESSAGES.noIpRules;
    },
  ),
  new ScenarioInput(
```

```
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      }
      return MESSAGES.noIpRules;
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    }
    return false;
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
```

```
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
  saveState,
];
```

Criar etapas para executar a demonstração.

```
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
```

```
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
```

```
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );

  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
```

```
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
```

```
...statusSteps,
new ScenarioInput(
  "brokenDependencyConfirmation",
  MESSAGES.demoBrokenDependencyConfirmation,
  { type: "confirm" },
),
new ScenarioAction("brokenDependency", async (state) => {
  if (!state.brokenDependencyConfirmation) {
    process.exit();
  } else {
    const client = new SSMClient({});
    state.badTableName = `fake-table-${Date.now()}`;
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: state.badTableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }
}),
new ScenarioOutput("testBrokenDependency", (state) =>
  MESSAGES.demoTestBrokenDependency.replace(
    "${TABLE_NAME}",
    state.badTableName,
  ),
),
...statusSteps,
new ScenarioInput(
  "staticResponseConfirmation",
  MESSAGES.demoStaticResponseConfirmation,
  { type: "confirm" },
),
new ScenarioAction("staticResponse", async (state) => {
  if (!state.staticResponseConfirmation) {
    process.exit();
  } else {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmFailureResponseKey,
        Value: "static",
        Overwrite: true,
```

```
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      const ec2Client = new EC2Client({});
```

```
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
```

```
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
```

```
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
```

```
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
```

```
      join(RESOURCES_PATH, "ssm_only_policy.json"),
    ),
  }),
);
await iamClient.send(
  new CreateRoleCommand({
    RoleName: NAMES.ssmOnlyRoleName,
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: { Service: "ec2.amazonaws.com" },
          Action: "sts:AssumeRole",
        },
      ],
    }),
  }),
);
await iamClient.send(
  new AttachRolePolicyCommand({
    RoleName: NAMES.ssmOnlyRoleName,
    PolicyArn: Policy.Arn,
  }),
);
await iamClient.send(
  new AttachRolePolicyCommand({
    RoleName: NAMES.ssmOnlyRoleName,
    PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
  }),
);
const { InstanceProfile } = await iamClient.send(
  new CreateInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
  }),
);
await waitUntilInstanceProfileExists(
  { client: iamClient },
  { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
await iamClient.send(
  new AddRoleToInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    RoleName: NAMES.ssmOnlyRoleName,
```

```
    }),
  );

  return InstanceProfile;
}
```

Criar etapas para destruir todos os recursos.

```
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
  RevokeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
```

```
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { loadState } from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  loadState,
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    }
    return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
```

```
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
    return MESSAGES.deletedKeyPair.replace(
      "${KEY_PAIR_NAME}",
      NAMES.keyPairName,
    );
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
```

```
    }
    return MESSAGES.detachedPolicyFromRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
    return MESSAGES.deletedPolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    );
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
```

```
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
    return MESSAGES.removedRoleFromInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
    return MESSAGES.deletedInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    );
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
```

```
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
    return MESSAGES.deletedInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    );
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
    return MESSAGES.deletedAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    );
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
```

```
    const client = new EC2Client({});
    try {
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
    return MESSAGES.deletedLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    );
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
```

```
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
    return MESSAGES.deletedLoadBalancer.replace(
      "${LB_NAME}",
      NAMES.loadBalancerName,
    );
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
    return MESSAGES.deletedLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    );
```

```
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
    return MESSAGES.detachedSsmOnlyRoleFromProfile
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
```

```
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
    return MESSAGES.detachedSsmOnlyCustomRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
    return MESSAGES.detachedSsmOnlyAWSRolePolicy
      .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
      .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
```

```
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
    return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.ssmOnlyInstanceProfileName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
    return MESSAGES.deletedSsmOnlyPolicy.replace(
      "${POLICY_NAME}",
      NAMES.ssmOnlyPolicyName,
    );
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
```

```
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
    return MESSAGES.deletedSsmOnlyRole.replace(
      "${ROLE_NAME}",
      NAMES.ssmOnlyRoleName,
    );
  }),
  new ScenarioAction(
    "revokeSecurityGroupIngress",
    async (
      /** @type {{ myIp: string, defaultSecurityGroup: { GroupId: string } }} */
state,
    ) => {
      const ec2Client = new EC2Client({});

      try {
        await ec2Client.send(
          new RevokeSecurityGroupIngressCommand({
            GroupId: state.defaultSecurityGroup.GroupId,
            CidrIp: `${state.myIp}/32`,
            FromPort: 80,
            ToPort: 80,
            IpProtocol: "tcp",
          }),
        );
      } catch (e) {
        state.revokeSecurityGroupIngressError = e;
      }
    },
  ),
  new ScenarioOutput("revokeSecurityGroupIngressResult", (state) => {
    if (state.revokeSecurityGroupIngressError) {
      console.error(state.revokeSecurityGroupIngressError);
```

```
      return MESSAGES.revokeSecurityGroupIngressError.replace(
        "${IP}",
        state.myIp,
      );
    }
    return MESSAGES.revokedSecurityGroupIngress.replace("${IP}", state.myIp);
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    }
    console.log(err.name);
    throw err;
  }
}

/**
```

```
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para JavaScript.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup

- CreateInstanceProfile
- CreateLaunchTemplate
- CreateListener
- CreateLoadBalancer
- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Execute o cenário interativo em um prompt de comando.

```
class Runner:
    """
    Manages the deployment, demonstration, and destruction of resources for the
 resilient service.
    """

    def __init__(
        self,
        resource_path: str,
        recommendation: RecommendationService,
        autoscaler: AutoScalingWrapper,
        loadbalancer: ElasticLoadBalancerWrapper,
        param_helper: ParameterHelper,
    ):
        """
        Initializes the Runner class with the necessary parameters.

        :param resource_path: The path to resource files used by this example,
 such as IAM policies and instance scripts.
        :param recommendation: An instance of the RecommendationService class.
        :param autoscaler: An instance of the AutoScaler class.
        :param loadbalancer: An instance of the LoadBalancer class.
        :param param_helper: An instance of the ParameterHelper class.
        """
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

        prefix = "doc-example-resilience"
        self.target_group_name = f"{prefix}-tg"
        self.load_balancer_name = f"{prefix}-lb"

    def deploy(self) -> None:
        """
        Deploys the resources required for the resilient service, including the
 DynamoDB table,
        EC2 instances, Auto Scaling group, and load balancer.
```

```
        """
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        logging.info("Starting deployment of resources for the resilient
service.")

        logging.info(
            "Creating and populating DynamoDB table '%s'.",
            self.recommendation.table_name,
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)

        logging.info(
            "Creating an EC2 launch template with the startup script '%s'.",
            startup_script,
        )
        self.autoscaler.create_template(startup_script, instance_policy)

        logging.info(
            "Creating an EC2 Auto Scaling group across multiple Availability
Zones."
        )
        zones = self.autoscaler.create_autoscaling_group(3)

        logging.info("Creating variables that control the flow of the demo.")
        self.param_helper.reset()

        logging.info("Creating Elastic Load Balancing target group and load
balancer.")

        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.target_group_name, self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            self.load_balancer_name, [subnet["SubnetId"] for subnet in subnets]
        )
        self.loadbalancer.create_listener(self.load_balancer_name, target_group)

        self.autoscaler.attach_load_balancer_target_group(target_group)
```

```
        logging.info("Verifying access to the load balancer endpoint.")
        endpoint = self.loadbalancer.get_endpoint(self.load_balancer_name)
        lb_success = self.loadbalancer.verify_load_balancer_endpoint(endpoint)
        current_ip_address = requests.get("http://
checkip.amazonaws.com").text.strip()

        if not lb_success:
            logging.warning(
                "Couldn't connect to the load balancer. Verifying that the port
 is open..."
            )
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                logging.warning(
                    "The default security group for your VPC must allow access
 from this computer."
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
 from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
```

```
            lb_success =
self.loadbalancer.verify_load_balancer_endpoint(endpoint)

        if lb_success:
            logging.info(
                "Load balancer is ready. Access it at: http://%s",
current_ip_address
            )
        else:
            logging.error(
                "Couldn't get a successful response from the load balancer
endpoint. Please verify your VPC and security group settings."
            )

    def demo_choices(self) -> None:
        """
        Presents choices for interacting with the deployed service, such as
sending requests to
        the load balancer or checking the health of the targets.
        """
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            logging.info("Choose an action to interact with the service.")
            choice = q.choose("Which action would you like to take? ", actions)
            if choice == 0:
                logging.info("Sending a GET request to the load balancer
endpoint.")
                endpoint =
self.loadbalancer.get_endpoint(self.load_balancer_name)
                logging.info("GET http://%s", endpoint)
                response = requests.get(f"http://{endpoint}")
                logging.info("Response: %s", response.status_code)
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                logging.info("Checking the health of load balancer targets.")
                health =
self.loadbalancer.check_target_health(self.target_group_name)
                for target in health:
```

```
                    state = target["TargetHealth"]["State"]
                    logging.info(
                        "Target %s on port %d is %s",
                        target["Target"]["Id"],
                        target["Target"]["Port"],
                        state,
                    )
                    if state != "healthy":
                        logging.warning(
                            "%s: %s",
                            target["TargetHealth"]["Reason"],
                            target["TargetHealth"]["Description"],
                        )
                logging.info(
                    "Note that it can take a minute or two for the health check
to update."
                )
            elif choice == 2:
                logging.info("Proceeding to the next part of the demo.")

    def demo(self) -> None:
        """
        Runs the demonstration, showing how the service responds to different
failure scenarios
        and how a resilient architecture can keep the service running.
        """
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        logging.info("Resetting parameters to starting values for the demo.")
        self.param_helper.reset()

        logging.info(
            "Starting demonstration of the service's resilience under various
failure conditions."
        )
        self.demo_choices()

        logging.info(
            "Simulating failure by changing the Systems Manager parameter to a
non-existent table."
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info("Sending GET requests will now return failure codes.")
        self.demo_choices()
```

```
        logging.info("Switching to static response mode to mitigate failure.")
        self.param_helper.put(self.param_helper.failure_response, "static")
        logging.info("Sending GET requests will now return static responses.")
        self.demo_choices()

        logging.info("Restoring normal operation of the recommendation service.")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)

        logging.info(
            "Introducing a failure by assigning bad credentials to one of the
instances."
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        logging.info(
            "Replacing instance profile with bad credentials for instance %s.",
            bad_instance_id,
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        logging.info(
            "Sending GET requests may return either a valid recommendation or a
static response."
        )
        self.demo_choices()

        logging.info("Implementing deep health checks to detect unhealthy
instances.")
        self.param_helper.put(self.param_helper.health_check, "deep")
        logging.info("Checking the health of the load balancer targets.")
        self.demo_choices()
```

```
        logging.info(
            "Terminating the unhealthy instance to let the auto scaler replace
it."
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        logging.info("The service remains resilient during instance
replacement.")
        self.demo_choices()

        logging.info("Simulating a complete failure of the recommendation
service.")
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        logging.info(
            "All instances will report as unhealthy, but the service will still
return static responses."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self, automation=False) -> None:
        """
        Destroys all resources created for the demo, including the load balancer,
Auto Scaling group,
        EC2 instances, and DynamoDB table.
        """
        logging.info(
            "This concludes the demo. Preparing to clean up all AWS resources
created during the demo."
        )
        if automation:
            cleanup = True
        else:
            cleanup = q.ask(
                "Do you want to clean up all demo resources? (y/n) ", q.is_yesno
            )

        if cleanup:
            logging.info("Deleting load balancer and related resources.")
            self.loadbalancer.delete_load_balancer(self.load_balancer_name)
            self.loadbalancer.delete_target_group(self.target_group_name)
            self.autoscaler.delete_autoscaling_group(self.autoscaler.group_name)
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
```

```
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            logging.info("Deleting DynamoDB table and other resources.")
            self.recommendation.destroy()
        else:
            logging.warning(
                "Resources have not been deleted. Ensure you clean them up
 manually to avoid unexpected charges."
            )


def main() -> None:
    """
    Main function to parse arguments and run the appropriate actions for the
 demo.
    """
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../scenarios/features/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    logging.info("Starting the Resilient Service demo.")

    prefix = "doc-example-resilience"

    # Service Clients
    ddb_client = boto3.client("dynamodb")
    elb_client = boto3.client("elbv2")
    autoscaling_client = boto3.client("autoscaling")
```

```
    ec2_client = boto3.client("ec2")
    ssm_client = boto3.client("ssm")
    iam_client = boto3.client("iam")

    # Wrapper instantiations
    recommendation = RecommendationService(
        "doc-example-recommendation-service", ddb_client
    )
    autoscaling_wrapper = AutoScalingWrapper(
        prefix,
        "t3.micro",
        "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    )
    elb_wrapper = ElasticLoadBalancerWrapper(elb_client)
    param_helper = ParameterHelper(recommendation.table_name, ssm_client)

    # Demo invocation
    runner = Runner(
        args.resource_path,
        recommendation,
        autoscaling_wrapper,
        elb_wrapper,
        param_helper,
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    logging.info("Demo completed successfully.")


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Crie uma classe que envolva ações do Auto Scaling e do Amazon EC2.

```
class AutoScalingWrapper:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix: str,
        inst_type: str,
        ami_param: str,
        autoscaling_client: boto3.client,
        ec2_client: boto3.client,
        ssm_client: boto3.client,
        iam_client: boto3.client,
    ):
        """
        Initializes the AutoScaler class with the necessary parameters.

        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        sts_client = boto3.client("sts")
        self.account_id = sts_client.get_caller_identity()["Account"]

        self.key_pair_name = f"{resource_prefix}-key-pair"
        self.launch_template_name = f"{resource_prefix}-template-"
```

```
        self.group_name = f"{resource_prefix}-group"

        # Happy path
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"

        # Failure mode
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"


    def create_policy(self, policy_file: str, policy_name: str) -> str:
        """
        Creates a new IAM policy or retrieves the ARN of an existing policy.

        :param policy_file: The path to a JSON file that contains the policy
 definition.
        :param policy_name: The name to give the created policy.
        :return: The ARN of the created or existing policy.
        """
        with open(policy_file) as file:
            policy_doc = file.read()

        try:
            response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=policy_doc
            )
            policy_arn = response["Policy"]["Arn"]
            log.info(f"Policy '{policy_name}' created successfully. ARN:
 {policy_arn}")
            return policy_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the policy already exists, get its ARN
                response = self.iam_client.get_policy(
                    PolicyArn=f"arn:aws:iam::{self.account_id}:policy/
{policy_name}"
                )
                policy_arn = response["Policy"]["Arn"]
                log.info(f"Policy '{policy_name}' already exists. ARN:
 {policy_arn}")
```

```
                return policy_arn
            log.error(f"Full error:\n\t{err}")

    def create_role(self, role_name: str, assume_role_doc: dict) -> str:
        """
        Creates a new IAM role or retrieves the ARN of an existing role.

        :param role_name: The name to give the created role.
        :param assume_role_doc: The assume role policy document that specifies
 which
                                entities can assume the role.
        :return: The ARN of the created or existing role.
        """
        try:
            response = self.iam_client.create_role(
                RoleName=role_name,
 AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            role_arn = response["Role"]["Arn"]
            log.info(f"Role '{role_name}' created successfully. ARN: {role_arn}")
            return role_arn

        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                # If the role already exists, get its ARN
                response = self.iam_client.get_role(RoleName=role_name)
                role_arn = response["Role"]["Arn"]
                log.info(f"Role '{role_name}' already exists. ARN: {role_arn}")
                return role_arn
            log.error(f"Full error:\n\t{err}")

    def attach_policy(
        self,
        role_name: str,
        policy_arn: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> None:
        """
        Attaches an IAM policy to a role and optionally attaches additional AWS-
managed policies.

        :param role_name: The name of the role to attach the policy to.
        :param policy_arn: The ARN of the policy to attach.
```

```
        :param aws_managed_policies: A tuple of AWS-managed policy names to
attach to the role.
        """
        try:
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info(f"Attached policy {policy_arn} to role {role_name}.")
        except ClientError as err:
            log.error(f"Failed to attach policy {policy_arn} to role
{role_name}.")
            log.error(f"Full error:\n\t{err}")

    def create_instance_profile(
        self,
        policy_file: str,
        policy_name: str,
        role_name: str,
        profile_name: str,
        aws_managed_policies: Tuple[str, ...] = (),
    ) -> str:
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
```

```
                                    use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version":"2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        policy_arn = self.create_policy(policy_file, policy_name)
        self.create_role(role_name, assume_role_doc)
        self.attach_policy(role_name, policy_arn, aws_managed_policies)

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            log.error(f"Full error:\n\t{err}")
        return profile_arn
```

```
   def get_instance_profile(self, instance_id: str) -> Dict[str, Any]:
       """
       Gets data about the profile associated with an instance.

       :param instance_id: The ID of the instance to look up.
       :return: The profile data.
       """
       try:
           response =
self.ec2_client.describe_iam_instance_profile_associations(
               Filters=[{"Name": "instance-id", "Values": [instance_id]}]
           )
           if not response["IamInstanceProfileAssociations"]:
               log.info(f"No instance profile found for instance
{instance_id}.")
           profile_data = response["IamInstanceProfileAssociations"][0]
           log.info(f"Retrieved instance profile for instance {instance_id}.")
           return profile_data
       except ClientError as err:
           log.error(
               f"Failed to retrieve instance profile for instance
{instance_id}."
           )
           error_code = err.response["Error"]["Code"]
           if error_code == "InvalidInstanceID.NotFound":
               log.error(f"The instance ID '{instance_id}' does not exist.")
           log.error(f"Full error:\n\t{err}")


   def replace_instance_profile(
       self,
       instance_id: str,
       new_instance_profile_name: str,
       profile_association_id: str,
   ) -> None:
       """
       Replaces the profile associated with a running instance. After the
profile is
       replaced, the instance is rebooted to ensure that it uses the new
profile. When
       the instance is ready, Systems Manager is used to restart the Python web
server.
```

```
        :param instance_id: The ID of the instance to restart.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)

            self.ec2_client.reboot_instances(InstanceIds=[instance_id])
            log.info("Rebooting instance %s.", instance_id)
            waiter = self.ec2_client.get_waiter("instance_running")
            log.info("Waiting for instance %s to be running.", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info("Instance %s is now running.", instance_id)

            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info(f"Restarted the Python web server on instance
'{instance_id}'.")
        except ClientError as err:
            log.error("Failed to replace instance profile.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidAssociationID.NotFound":
                log.error(
                    f"Association ID '{profile_association_id}' does not exist."
                    "Please check the association ID and try again."
                )
            if error_code == "InvalidInstanceId":
                log.error(
```

```
                    f"The specified instance ID '{instance_id}' does not exist or
is not available for SSM. "
                    f"Please verify the instance ID and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_instance_profile(self, profile_name: str, role_name: str) -> None:
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            log.error(
                f"Couldn't delete instance profile {profile_name} or detach "
                f"policies and delete role {role_name}: {err}"
            )
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
```

```
                )


    def create_key_pair(self, key_pair_name: str) -> None:
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to create key pair {key_pair_name}.")
            if error_code == "InvalidKeyPair.Duplicate":
                log.error(f"A key pair with the name '{key_pair_name}' already
exists.")
            log.error(f"Full error:\n\t{err}")


    def delete_key_pair(self) -> None:
        """
        Deletes a key pair.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            log.error(f"Couldn't delete key pair '{self.key_pair_name}'.")
            log.error(f"Full error:\n\t{err}")
        except FileNotFoundError as err:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
            log.error(f"Full error:\n\t{err}")


    def create_template(
        self, server_startup_script_file: str, instance_policy_file: str
    ) -> Dict[str, Any]:
```

```
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            # Create key pair and instance profile
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )

            # Read the startup script
            with open(server_startup_script_file) as file:
                start_server_script = file.read()

            # Get the latest AMI ID
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]

            # Create the launch template
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
```

```
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                f"Created launch template {self.launch_template_name} for AMI
{ami_id} on {self.inst_type}."
            )
        except ClientError as err:
            log.error(f"Failed to create launch template
{self.launch_template_name}.")
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidLaunchTemplateName.AlreadyExistsException":
                log.info(
                    f"Launch template {self.launch_template_name} already exists,
nothing to do."
                )
            log.error(f"Full error:\n\t{err}")
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
```

```
                )
            log.error(f"Full error:\n\t{err}")


    def get_availability_zones(self) -> List[str]:
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
            log.info(f"Retrieved {len(zones)} availability zones: {zones}.")
        except ClientError as err:
            log.error("Failed to retrieve availability zones.")
            log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def create_autoscaling_group(self, group_size: int) -> List[str]:
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
```

```
                f"Created EC2 Auto Scaling group {self.group_name} with
availability zones {zones}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            if error_code == "AlreadyExists":
                log.info(
                    f"EC2 Auto Scaling group {self.group_name} already exists,
nothing to do."
                )
            else:
                log.error(f"Failed to create EC2 Auto Scaling group
{self.group_name}.")
                log.error(f"Full error:\n\t{err}")
        else:
            return zones


    def get_instances(self) -> List[str]:
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: A list of instance IDs in the Auto Scaling group.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
            log.info(
                f"Retrieved {len(instance_ids)} instances for Auto Scaling group
{self.group_name}."
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to retrieve instances for Auto Scaling group
{self.group_name}."
            )
            if error_code == "ResourceNotFound":
```

```
                log.error(f"The Auto Scaling group '{self.group_name}' does not
exist.")
            log.error(f"Full error:\n\t{err}")
        else:
            return instance_ids


    def terminate_instance(self, instance_id: str, decrementsetting=False) ->
None:
        """
        Terminates an instance in an EC2 Auto Scaling group. After an instance is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        :param decrementsetting: If True, do not replace terminated instances.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id,
                ShouldDecrementDesiredCapacity=decrementsetting,
            )
            log.info("Terminated instance %s.", instance_id)

            # Adding a waiter to ensure the instance is terminated
            waiter = self.ec2_client.get_waiter("instance_terminated")
            log.info("Waiting for instance %s to be terminated...", instance_id)
            waiter.wait(InstanceIds=[instance_id])
            log.info(
                f"Instance '{instance_id}' has been terminated and will be
replaced."
            )

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to terminate instance '{instance_id}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
to terminate the instance again."
                )
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
```

```
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            log.error(f"Full error:\n\t{err}")

    def attach_load_balancer_target_group(
        self, lb_target_group: Dict[str, Any]
    ) -> None:
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forwards requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to attach load balancer target group
'{lb_target_group['TargetGroupName']}'."
            )
            if error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
the resource."
                )
            elif error_code == "ServiceLinkedRoleFailure":
                log.error(
                    "The operation failed because the service-linked role is not
ready or does not exist. "
```

```
                    "Check that the service-linked role exists and is correctly
 configured."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_autoscaling_group(self, group_name: str) -> None:
        """
        Terminates all instances in the group, then deletes the EC2 Auto Scaling
 group.

        :param group_name: The name of the group to delete.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self.terminate_instance(inst_id)

                # Wait for all instances to be terminated
                if instance_ids:
                    waiter = self.ec2_client.get_waiter("instance_terminated")
                    log.info("Waiting for all instances to be terminated...")
                    waiter.wait(InstanceIds=instance_ids)
                    log.info("All instances have been terminated.")
            else:
                log.info(f"No groups found named '{group_name}'! Nothing to do.")
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete Auto Scaling group '{group_name}'.")
            if error_code == "ScalingActivityInProgressFault":
                log.error(
                    "Scaling activity is currently in progress. "
                    "Wait for the scaling activity to complete before attempting
 to delete the group again."
                )
```

```
            elif error_code == "ResourceContentionFault":
                log.error(
                    "The request failed due to a resource contention issue. "
                    "Ensure that no conflicting operations are being performed on
 the group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_default_vpc(self) -> Dict[str, Any]:
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error("Failed to retrieve the default VPC.")
            if error_code == "UnauthorizedOperation":
                log.error(
                    "You do not have the necessary permissions to describe VPCs.
 "
                    "Ensure that your AWS IAM user or role has the correct
 permissions."
                )
            elif error_code == "InvalidParameterValue":
                log.error(
                    "One or more parameters are invalid. Check the request
 parameters."
                )

            log.error(f"Full error:\n\t{err}")
        else:
            if "Vpcs" in response and response["Vpcs"]:
                log.info(f"Retrieved default VPC: {response['Vpcs'][0]
['VpcId']}")
                return response["Vpcs"][0]
            else:
                pass
```

```
   def verify_inbound_port(
       self, vpc: Dict[str, Any], port: int, ip_address: str
   ) -> Tuple[Dict[str, Any], bool]:
       """
       Verify the default security group of the specified VPC allows ingress
from this
       computer. This can be done by allowing ingress from this computer's IP
       address. In some situations, such as connecting from a corporate network,
you
       must instead specify a prefix list ID. You can also temporarily open the
port to
       any IP address while running this example. If you do, be sure to remove
public
       access when you're done.

       :param vpc: The VPC used by this example.
       :param port: The port to verify.
       :param ip_address: This computer's IP address.
       :return: The default security group of the specified VPC, and a value
that indicates
                whether the specified port is open.
       """
       try:
           response = self.ec2_client.describe_security_groups(
               Filters=[
                   {"Name": "group-name", "Values": ["default"]},
                   {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
               ]
           )
           sec_group = response["SecurityGroups"][0]
           port_is_open = False
           log.info(f"Found default security group {sec_group['GroupId']}.")

           for ip_perm in sec_group["IpPermissions"]:
               if ip_perm.get("FromPort", 0) == port:
                   log.info(f"Found inbound rule: {ip_perm}")
                   for ip_range in ip_perm["IpRanges"]:
                       cidr = ip_range.get("CidrIp", "")
                       if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                           port_is_open = True
                   if ip_perm["PrefixListIds"]:
                       port_is_open = True
                   if not port_is_open:
```

```
                        log.info(
                            f"The inbound rule does not appear to be open to
either this computer's IP "
                            f"address of {ip_address}, to all IP addresses
(0.0.0.0/0), or to a prefix list ID."
                        )
                    else:
                        break
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to verify inbound rule for port {port} for VPC
{vpc['VpcId']}."
            )
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    f"The specified VPC ID '{vpc['VpcId']}' does not exist.
Please check the VPC ID."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id: str, port: int, ip_address: str) ->
None:
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
```

```
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to authorize ingress to security group '{sec_group_id}'
on port {port} from {ip_address}."
            )
            if error_code == "InvalidGroupId.Malformed":
                log.error(
                    "The security group ID is malformed. "
                    "Please verify that the security group ID is correct."
                )
            elif error_code == "InvalidPermission.Duplicate":
                log.error(
                    "The specified rule already exists in the security group. "
                    "Check the existing rules for this security group."
                )
            log.error(f"Full error:\n\t{err}")


    def get_subnets(self, vpc_id: str, zones: List[str] = None) -> List[Dict[str,
Any]]:
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        # Ensure that 'zones' is a list, even if None is passed
        if zones is None:
            zones = []
        try:
            paginator = self.ec2_client.get_paginator("describe_subnets")
            page_iterator = paginator.paginate(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
```

```
                ]
            )

            subnets = []
            for page in page_iterator:
                subnets.extend(page["Subnets"])

            log.info("Found %s subnets for the specified zones.", len(subnets))
            return subnets
        except ClientError as err:
            log.error(
                f"Failed to retrieve subnets for VPC '{vpc_id}' in zones
 {zones}."
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "InvalidVpcID.NotFound":
                log.error(
                    "The specified VPC ID does not exist. "
                    "Please check the VPC ID and try again."
                )
            # Add more error-specific handling as needed
            log.error(f"Full error:\n\t{err}")
```

Crie uma classe que envolva ações do Elastic Load Balancing.

```
class ElasticLoadBalancerWrapper:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, elb_client: boto3.client):
        """
        Initializes the LoadBalancer class with the necessary parameters.
        """
        self.elb_client = elb_client


    def create_target_group(
        self, target_group_name: str, protocol: str, port: int, vpc_id: str
    ) -> Dict[str, Any]:
        """
```

```
        Creates an Elastic Load Balancing target group. The target group
specifies how
        the load balancer forwards requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param target_group_name: The name of the target group to create.
        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info(f"Created load balancing target group
'{target_group_name}'.")
            return target_group
        except ClientError as err:
            log.error(
                f"Couldn't create load balancing target group
'{target_group_name}'."
            )
            error_code = err.response["Error"]["Code"]

            if error_code == "DuplicateTargetGroupName":
                log.error(
                    f"Target group name {target_group_name} already exists. "
```

```
                    "Check if the target group already exists."
                    "Consider using a different name or deleting the existing
target group if appropriate."
                )
            elif error_code == "TooManyTargetGroups":
                log.error(
                    "Too many target groups exist in the account. "
                    "Consider deleting unused target groups to create space for
new ones."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_target_group(self, target_group_name) -> None:
        """
        Deletes the target group.
        """
        try:
            # Describe the target group to get its ARN
            response =
self.elb_client.describe_target_groups(Names=[target_group_name])
            tg_arn = response["TargetGroups"][0]["TargetGroupArn"]

            # Delete the target group
            self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
            log.info("Deleted load balancing target group %s.",
target_group_name)

            # Use a custom waiter to wait until the target group is no longer
available
            self.wait_for_target_group_deletion(self.elb_client, tg_arn)
            log.info("Target group %s successfully deleted.", target_group_name)

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to delete target group '{target_group_name}'.")
            if error_code == "TargetGroupNotFound":
                log.error(
                    "Load balancer target group either already deleted or never
existed. "
                    "Verify the name and check that the resource exists in the
AWS Console."
                )
            elif error_code == "ResourceInUseException":
```

```
                log.error(
                    "Target group still in use by another resource. "
                    "Ensure that the target group is no longer associated with
any load balancers or resources.",
                )
            log.error(f"Full error:\n\t{err}")

    def wait_for_target_group_deletion(
        self, elb_client, target_group_arn, max_attempts=10, delay=30
    ):
        for attempt in range(max_attempts):
            try:

elb_client.describe_target_groups(TargetGroupArns=[target_group_arn])
                print(
                    f"Attempt {attempt + 1}: Target group {target_group_arn}
still exists."
                )
            except ClientError as e:
                if e.response["Error"]["Code"] == "TargetGroupNotFound":
                    print(
                        f"Target group {target_group_arn} has been successfully
deleted."
                    )
                    return
                else:
                    raise
            time.sleep(delay)
        raise TimeoutError(
            f"Target group {target_group_arn} was not deleted after {max_attempts
* delay} seconds."
        )


    def create_load_balancer(
        self,
        load_balancer_name: str,
        subnet_ids: List[str],
    ) -> Dict[str, Any]:
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.
```

```
        :param load_balancer_name: The name of the load balancer to create.
        :param subnet_ids: A list of subnets to associate with the load balancer.
        :return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info(f"Created load balancer '{load_balancer_name}'.")

            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info(
                f"Waiting for load balancer '{load_balancer_name}' to be
available..."
            )
            waiter.wait(Names=[load_balancer_name])
            log.info(f"Load balancer '{load_balancer_name}' is now available!")

        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to create load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "DuplicateLoadBalancerNameException":
                log.error(
                    f"A load balancer with the name '{load_balancer_name}'
already exists. "
                    "Load balancer names must be unique within the AWS region. "
                    "Please choose a different name and try again."
                )
            if error_code == "TooManyLoadBalancersException":
                log.error(
                    "The maximum number of load balancers has been reached in
this account and region. "
                    "You can delete unused load balancers or request an increase
in the service quota from AWS Support."
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return load_balancer
```

```
   def create_listener(
       self,
       load_balancer_name: str,
       target_group: Dict[str, Any],
   ) -> Dict[str, Any]:
       """
       Creates a listener for the specified load balancer that forwards requests
to the
       specified target group.

       :param load_balancer_name: The name of the load balancer to create a
listener for.
       :param target_group: An existing target group that is added as a listener
to the
                            load balancer.
       :return: Data about the newly created listener.
       """
       try:
           # Retrieve the load balancer ARN
           load_balancer_response = self.elb_client.describe_load_balancers(
               Names=[load_balancer_name]
           )
           load_balancer_arn = load_balancer_response["LoadBalancers"][0][
               "LoadBalancerArn"
           ]

           # Create the listener
           response = self.elb_client.create_listener(
               LoadBalancerArn=load_balancer_arn,
               Protocol=target_group["Protocol"],
               Port=target_group["Port"],
               DefaultActions=[
                   {
                       "Type": "forward",
                       "TargetGroupArn": target_group["TargetGroupArn"],
                   }
               ],
           )
           log.info(
               f"Created listener to forward traffic from load balancer
'{load_balancer_name}' to target group '{target_group['TargetGroupName']}'."
           )
           return response["Listeners"][0]
```

```
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
                f"Failed to add a listener on '{load_balancer_name}' for target
group '{target_group['TargetGroupName']}'."
            )

            if error_code == "ListenerNotFoundException":
                log.error(
                    f"The listener could not be found for the load balancer
'{load_balancer_name}'. "
                    "Please check the load balancer name and target group
configuration."
                )
            if error_code == "InvalidConfigurationRequestException":
                log.error(
                    f"The configuration provided for the listener on load
balancer '{load_balancer_name}' is invalid. "
                    "Please review the provided protocol, port, and target group
settings."
                )
            log.error(f"Full error:\n\t{err}")


    def delete_load_balancer(self, load_balancer_name) -> None:
        """
        Deletes a load balancer.

        :param load_balancer_name: The name of the load balancer to delete.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[load_balancer_name])
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(
```

```
                f"Couldn't delete load balancer '{load_balancer_name}'. Error
code: {error_code}, Message: {err.response['Error']['Message']}"
            )

            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    f"The load balancer '{load_balancer_name}' does not exist. "
                    "Please check the name and try again."
                )
            log.error(f"Full error:\n\t{err}")


    def get_endpoint(self, load_balancer_name) -> str:
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[load_balancer_name]
            )
            return response["LoadBalancers"][0]["DNSName"]
        except ClientError as err:
            log.error(
                f"Couldn't get the endpoint for load balancer
{load_balancer_name}"
            )
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Verify load balancer name and ensure it exists in the AWS
console."
                )
            log.error(f"Full error:\n\t{err}")

    @staticmethod
    def verify_load_balancer_endpoint(endpoint) -> bool:
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.

        :param endpoint: The endpoint to verify.
        :return: True if the GET request is successful, False otherwise.
```

```
        """
        retries = 3
        verified = False
        while not verified and retries > 0:
            try:
                lb_response = requests.get(f"http://{endpoint}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    verified = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
                )
                retries -= 1
                time.sleep(10)
        return verified

    def check_target_health(self, target_group_name: str) -> List[Dict[str,
Any]]:
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            log.error(f"Couldn't check health of {target_group_name} target(s).")
            error_code = err.response["Error"]["Code"]
            if error_code == "LoadBalancerNotFoundException":
                log.error(
                    "Load balancer associated with the target group was not
found. "
```

```
                    "Ensure the load balancer exists, is in the correct AWS
 region, and "
                    "that you have the necessary permissions to access it.",
                )
            elif error_code == "TargetGroupNotFoundException":
                log.error(
                    "Target group was not found. "
                    "Verify the target group name, check that it exists in the
 correct region, "
                    "and ensure it has not been deleted or created in a different
 account.",
                )
            log.error(f"Full error:\n\t{err}")
        else:
            return health_response["TargetHealthDescriptions"]
```

Crie uma classe que use o DynamoDB para simular um serviço de recomendação.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name: str, dynamodb_client: boto3.client):
        """
        Initializes the RecommendationService class with the necessary
 parameters.

        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    def create(self) -> Dict[str, Any]:
        """
```

```
        Creates a DynamoDB table to use as a recommendation service. The table
has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        :raises RecommendationServiceError: If the table creation fails.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be done.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file: str) -> None:
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
```

```
        :raises RecommendationServiceError: If the table population fails.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self) -> None:
        """
        Deletes the recommendations table.

        :raises RecommendationServiceError: If the table deletion fails.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Crie uma classe que envolva as ações do Systems Manager.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table: str = "doc-example-resilient-architecture-table"
    failure_response: str = "doc-example-resilient-architecture-failure-response"
    health_check: str = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name: str, ssm_client: boto3.client):
        """
        Initializes the ParameterHelper class with the necessary parameters.

        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    def reset(self) -> None:
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
 a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name: str, value: str) -> None:
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        :raises ParameterHelperError: If the parameter value cannot be set.
```

```
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting parameter %s to '%s'.", name, value)
        except ClientError as err:
            error_code = err.response["Error"]["Code"]
            log.error(f"Failed to set parameter {name}.")
            if error_code == "ParameterLimitExceeded":
                log.error(
                    "The parameter limit has been exceeded. "
                    "Consider deleting unused parameters or request a limit
increase."
                )
            elif error_code == "ParameterAlreadyExists":
                log.error(
                    "The parameter already exists and overwrite is set to False.
"
                    "Use Overwrite=True to update the parameter."
                )
            log.error(f"Full error:\n\t{err}")
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer

- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Configurar a conectividade do serviço de contêiner

O código de exemplo abaixo mostra como:

- Criar a infraestrutura da VPC.
- Configurar registro em log
- Criar o cluster do ECS.
- Configurar perfis do IAM.
- Criar o serviço com o Service Connect.
- Verificar a implantação
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
```

```
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"
```

```
    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"
```

```
    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true
```

```
    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
```

```
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
```

```
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."
```

```
    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"
```

```
    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
```

```
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"
```

```
    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi
```

```
    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "=========================================="
    echo "CREATED RESOURCES"
    echo "=========================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "=========================================="
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
```

```
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
```

```
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups
```

```
    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "============================================"
    echo "CLEANUP CONFIRMATION"
    echo "============================================"
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
```

```
main "$@"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition
  - DescribeClusters
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNamespaces
  - ListServices
  - ListTaskDefinitions
  - ListTasks

- RegisterTaskDefinition
- UpdateService
- Aguardar (Amazon EC2)
- Aguardar (Amazon ECS)

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor

O código de exemplo abaixo mostra como:

- Criar um cluster.
- Criar uma definição de tarefa.
- Criar o serviço.
- Fazer a limpeza.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error
```

```
# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
    local description="$2"

    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "=========================================="
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "=========================================="

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
```

```
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
```

```
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
```

```
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "============================================="
    echo "CLEANUP PROCESS"
    echo "============================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "Auto-confirming cleanup of all created resources..."
```

```
   CLEANUP_CHOICE="y"

   if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
       echo "Starting cleanup process..."

       # Step 1: Scale service to 0 tasks first, then delete service
       if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
then
           echo ""
           echo "Step 1: Scaling service to 0 tasks..."
           if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
               echo "Waiting for service to stabilize after scaling to 0..."
               execute_command "aws ecs wait services-stable --cluster
'$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

               echo "Deleting service..."
               execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME'" "Delete ECS service"
           else
               echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
               execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
           fi
       fi

       # Step 2: Wait a bit for tasks to fully terminate
       echo ""
       echo "Step 2: Waiting for tasks to fully terminate..."
       sleep 15

       # Step 3: Delete cluster
       if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
           echo ""
           echo "Step 3: Deleting cluster..."
           execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
       fi

       # Step 4: Wait for network interfaces to be cleaned up, then delete
security group
```

```
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
            local revisions
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "============================================"
        echo "CLEANUP COMPLETED"
        echo "============================================"
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
```

```
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}

# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "==========================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "==========================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
```

```
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
```

```
echo ""
echo "============================================="
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================="

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================="
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================="

# Create task definition JSON with validated inputs
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
```

```
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "============================================="
echo "STEP 4: SET UP NETWORKING"
echo "============================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
```

```
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
```

```
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "=========================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "=========================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")
```

```
# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "============================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "============================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"

# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
```

```
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "=========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "=========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "=========================================="
echo "SERVICE INFORMATION"
echo "=========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "=========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "=========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.

- AttachRolePolicy
- AuthorizeSecurityGroupIngress
- CreateCluster
- CreateRole
- CreateSecurityGroup
- CreateService
- DeleteCluster
- DeleteSecurityGroup
- DeleteService
- DeregisterTaskDefinition
- DescribeNetworkInterfaces
- DescribeSecurityGroups
- DescribeServices
- DescribeSubnets
- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetRole
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar uma API REST com integração de proxy de função

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM para execução do Lambda.

- Criar e implantar uma função do Lambda.
- Criar uma API REST
- Configurar a integração de proxy do Lambda.
- Implantar e testar a API
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."
```

```
# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
```

```
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration

# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
```

```
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
```

```
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}
```

```
echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."

# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2
```

```
# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddPermission
  - AttachRolePolicy
  - CreateDeployment
  - CreateFunction
  - CreateResource
  - CreateRestApi
  - CreateRole
  - DeleteFunction
  - DeleteRestApi
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetResources
  - GetRestApis
  - PutIntegration
  - PutMethod

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK

Os exemplos de código a seguir mostram como criar usuários e anexar políticas a eles.

> **Warning**
>
> Para evitar riscos de segurança, não use usuários do IAM para autenticação ao desenvolver software com propósito específico ou trabalhar com dados reais. Em vez disso, use federação com um provedor de identidade, como Centro de Identidade do AWS IAM.

- Criar dois usuários do IAM.
- Anexe uma política para um usuário obter e colocar objetos em um bucket do Amazon S3.
- Anexar uma política para o segundo usuário para obter objetos do bucket.
- Obter outras permissões para o bucket com base nas credenciais do usuário.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações do usuário do IAM.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
```

```
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user



def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
```

```
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
```

```
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

Crie a funções que envolvam ações de política do IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
```

```
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Crie funções que envolvam ações de chave de acesso do IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """
```

```
    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Use as funções de wrapper para criar usuários com políticas diferentes e use as credenciais deles para acessar um bucket do Amazon S3.

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
    This demonstration creates two users: one user who can put and get objects in
 an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
 permitted
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
```

```
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
```

```
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )

    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
```

```
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - AttachUserPolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteUser
  - DetachUserPolicy
  - ListUsers
  - UpdateUser

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar um serviço de contêiner para instâncias de máquinas virtuais

O código de exemplo abaixo mostra como:

- Criar um cluster do ECS.
- Criar e monitorar um serviço.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()
```

```
# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}

# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi
```

```
        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Deregister task definition
    if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
        log "Deregistering task definition: $TASK_DEFINITION_ARN"
        if ! aws ecs deregister-task-definition --task-definition
 "$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to deregister task definition
 $TASK_DEFINITION_ARN"
            ((cleanup_errors++))
        fi
    fi

    # Terminate EC2 instance
    if [[ -n "${INSTANCE_ID:-}" ]]; then
        log "Terminating EC2 instance: $INSTANCE_ID"
        if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
 2>>"$LOG_FILE"; then
            log "WARNING: Failed to terminate instance $INSTANCE_ID"
            ((cleanup_errors++))
```

```
        else
            log "Waiting for instance to terminate..."
            if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
                log "WARNING: Instance termination wait failed for $INSTANCE_ID"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Delete security group with retry logic
    if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
        log "Deleting security group: $SECURITY_GROUP_ID"
        local retry_count=0
        local max_retries=3

        while [[ $retry_count -lt $max_retries ]]; do
            if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
                log "Successfully deleted security group"
                break
            else
                ((retry_count++))
                if [[ $retry_count -lt $max_retries ]]; then
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
$max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
```

```
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"
```

```
    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
```

```
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
```

```
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
```

```
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole

        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."
```

```
    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
```

```
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
```

```
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi
```

```
    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"
```

```
    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun

    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
```

```
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "=========================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "=========================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
```

```
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
```

```
main "$@"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeContainerInstances
  - DescribeInstances
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - GetCallerIdentity
  - GetParameters
  - GetRole
  - ListContainerInstances
  - ListServices
  - ListTasks
  - RegisterTaskDefinition
  - RunInstances

- StopTask
- TerminateInstances
- UpdateService
- Aguardar (Amazon EC2)
- Aguardar (Amazon ECS)
- Aguardar (IAM)

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar um espaço de trabalho de monitoramento gerenciado

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM para seu espaço de trabalho.
- Criar um espaço de trabalho do Grafana.
- Configurar autenticação
- Definir configurações opcionais.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it
```

```
# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."

    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi
```

```
    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."

# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
```

```
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"
```

```
ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."
```

```
    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "==========================================="
echo "WORKSPACE INFORMATION"
echo "==========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
```

```
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "========================================="

# Step 6: Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
 grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done
```

```
    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
 still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace

- DescribeWorkspace
- DetachRolePolicy
- GetCallerIdentity

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar um painel de monitoramento com o nome da função como uma variável

O código de exemplo abaixo mostra como:

- Criar um painel do CloudWatch
- Adicionar widgets de métricas do Lambda com uma variável de nome de função.
- Verificar o painel.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1
```

```
echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "=========================================="
    echo "CLEANUP CONFIRMATION"
    echo "=========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true

    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi
```

```
# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."

    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi
```

```
    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
```

```
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
```

```
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
```

```
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
```

```
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
```

```
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"
```

```
if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi
```

```
    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetDashboard
  - Invocar
  - ListDashboards
  - ListFunctions
  - PutDashboard

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar sua primeira função com tecnologia sem servidor

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM para o Lambda.
- Criar código da função.
- Criar uma função do Lambda
- Testar uma função do Lambda.

- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash
# AWS Lambda - Create Your First Function
# This script creates a Lambda function, invokes it with a test event,
# views CloudWatch logs, and cleans up all resources.
#
# Source: https://docs.aws.amazon.com/lambda/latest/dg/getting-started.html
#
# Resources created:
#   - IAM role (Lambda execution role with basic logging permissions)
#   - Lambda function (Python 3.13 or Node.js 22.x runtime)
#   - CloudWatch log group (created automatically by Lambda on invocation)

set -eE -o pipefail

################################################################################
# Setup
################################################################################

UNIQUE_ID=$(head -c 8 /dev/urandom | od -An -tx1 | tr -d ' ')
FUNCTION_NAME="my-lambda-function-${UNIQUE_ID}"
ROLE_NAME="lambda-execution-role-${UNIQUE_ID}"
LOG_GROUP_NAME="/aws/lambda/${FUNCTION_NAME}"

TEMP_DIR=$(mktemp -d)
readonly TEMP_DIR
LOG_FILE="${TEMP_DIR}/lambda-gettingstarted.log"

exec > >(tee -a "$LOG_FILE") 2>&1

declare -a CREATED_RESOURCES
```

```
################################################################################
# Helper functions
################################################################################

cleanup_resources() {
    # Disable error trap to prevent recursion during cleanup
    trap - ERR
    set +eE

    echo ""
    echo "Cleaning up resources..."
    echo ""

    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        local RESOURCE="${CREATED_RESOURCES[$i]}"
        local TYPE="${RESOURCE%%:*}"
        local NAME="${RESOURCE#*:}"

        case "$TYPE" in
            log-group)
                echo "Deleting CloudWatch log group: ${NAME}"
                aws logs delete-log-group \
                    --log-group-name "$NAME" 2>&1 || echo "  WARNING: Could not
 delete log group ${NAME}."
                ;;
            lambda-function)
                echo "Deleting Lambda function: ${NAME}"
                aws lambda delete-function \
                    --function-name "$NAME" 2>&1 || echo "  WARNING: Could not
 delete Lambda function ${NAME}."
                echo "  Waiting for function deletion to complete..."
                local DELETE_WAIT=0
                while aws lambda get-function --function-name "$NAME" > /dev/null
 2>&1; do
                    sleep 2
                    DELETE_WAIT=$((DELETE_WAIT + 2))
                    if [ "$DELETE_WAIT" -ge 60 ]; then
                        echo "  WARNING: Timed out waiting for function
 deletion."
                        break
                    fi
                done
                ;;
```

```
            iam-role-policy)
                local ROLE_PART="${NAME%%|*}"
                local POLICY_PART="${NAME#*|}"
                echo "Detaching policy from role: ${ROLE_PART}"
                aws iam detach-role-policy \
                    --role-name "$ROLE_PART" \
                    --policy-arn "$POLICY_PART" 2>&1 || echo "  WARNING: Could
 not detach policy from role ${ROLE_PART}."
                ;;
            iam-role)
                echo "Deleting IAM role: ${NAME}"
                aws iam delete-role \
                    --role-name "$NAME" 2>&1 || echo "  WARNING: Could not delete
 IAM role ${NAME}."
                ;;
        esac
    done

    if [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi

    echo ""
    echo "Cleanup complete."
}

handle_error() {
    echo ""
    echo "========================================="
    echo "ERROR: Script failed at $1"
    echo "========================================="
    echo ""
    if [ ${#CREATED_RESOURCES[@]} -gt 0 ]; then
        echo "Attempting to clean up ${#CREATED_RESOURCES[@]} resource(s)..."
        cleanup_resources
    fi
    exit 1
}

trap 'handle_error "line $LINENO"' ERR

wait_for_resource() {
    local DESCRIPTION="$1"
    local COMMAND="$2"
```

```
    local TARGET_VALUE="$3"
    local TIMEOUT=300
    local ELAPSED=0
    local INTERVAL=5

    echo "Waiting for ${DESCRIPTION}..."
    while true; do
        local RESULT
        RESULT=$(eval "$COMMAND" 2>&1) || true
        if echo "$RESULT" | grep -q "$TARGET_VALUE"; then
            echo "  ${DESCRIPTION} is ready."
            return 0
        fi
        if [ "$ELAPSED" -ge "$TIMEOUT" ]; then
            echo "ERROR: Timed out waiting for ${DESCRIPTION} after ${TIMEOUT}
 seconds."
            return 1
        fi
        sleep "$INTERVAL"
        ELAPSED=$((ELAPSED + INTERVAL))
    done
}

validate_input() {
    local input="$1"
    local pattern="$2"
    if ! [[ "$input" =~ $pattern ]]; then
        echo "ERROR: Invalid input: $input"
        return 1
    fi
    return 0
}

################################################################################
# Region pre-check
################################################################################

CONFIGURED_REGION=$(aws configure get region 2>/dev/null || true)
if [ -z "$CONFIGURED_REGION" ] && [ -z "$AWS_DEFAULT_REGION" ] && [ -z
 "$AWS_REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Run 'aws configure set region <region>' or export AWS_DEFAULT_REGION."
    exit 1
fi
```

```
################################################################################
# Runtime selection
################################################################################

echo ""
echo "=========================================="
echo "AWS Lambda - Create Your First Function"
echo "=========================================="
echo ""
echo "Select a runtime for your Lambda function:"
echo "  1) Python 3.13"
echo "  2) Node.js 22.x"
echo ""
echo "Using default: Python 3.13"
RUNTIME_CHOICE="1"

case "$RUNTIME_CHOICE" in
    1)
        RUNTIME="python3.13"
        HANDLER="lambda_function.lambda_handler"
        CODE_FILE="lambda_function.py"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'PYTHON_EOF'
import json
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)
def lambda_handler(event, context):
    if not isinstance(event, dict) or 'length' not in event or 'width' not in
 event:
        raise ValueError('Event must contain length and width')
    try:
        length = float(event['length'])
        width = float(event['width'])
        if length < 0 or width < 0:
            raise ValueError('Length and width must be non-negative')
        area = calculate_area(length, width)
        print(f'The area is {area}')
        logger.info(f'CloudWatch logs group: {context.log_group_name}')
        return json.dumps({'area': area})
    except (TypeError, ValueError) as e:
        logger.error(f'Error processing input: {str(e)}')
        raise
def calculate_area(length, width):
```

```
    return length * width
PYTHON_EOF
        echo "Selected runtime: Python 3.13"
        ;;
    2)
        RUNTIME="nodejs22.x"
        HANDLER="index.handler"
        CODE_FILE="index.mjs"
        cat > "${TEMP_DIR}/${CODE_FILE}" << 'NODEJS_EOF'
export const handler = async (event, context) => {
  if (!event || typeof event.length !== 'number' || typeof event.width !==
 'number') {
    throw new Error('Event must contain numeric length and width');
  }
  if (event.length < 0 || event.width < 0) {
    throw new Error('Length and width must be non-negative');
  }
  const area = event.length * event.width;
  console.log(`The area is ${area}`);
  console.log('CloudWatch log group: ', context.logGroupName);
  return JSON.stringify({area});
};
NODEJS_EOF
        echo "Selected runtime: Node.js 22.x"
        ;;
    *)
        echo "ERROR: Invalid choice. Please enter 1 or 2."
        exit 1
        ;;
esac

# ============================================================================
# Step 1: Create IAM execution role
# ============================================================================

echo ""
echo "=========================================="
echo "Step 1: Create IAM execution role"
echo "=========================================="
echo ""

TRUST_POLICY='{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lambda.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}'

echo "Creating IAM role: ${ROLE_NAME}"
ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" \
    --query 'Role.Arn' \
    --output text 2>&1)

if ! validate_input "$ROLE_OUTPUT" "^arn:aws:iam::[0-9]+:role/"; then
    echo "ERROR: Failed to create IAM role"
    exit 1
fi

echo "$ROLE_OUTPUT"
ROLE_ARN="$ROLE_OUTPUT"
CREATED_RESOURCES+=("iam-role:${ROLE_NAME}")
echo "Role ARN: ${ROLE_ARN}"

aws iam tag-role \
    --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=lambda-gettingstarted

echo ""
echo "Attaching AWSLambdaBasicExecutionRole policy..."
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>&1
CREATED_RESOURCES+=("iam-role-policy:${ROLE_NAME}|arn:aws:iam::aws:policy/
service-role/AWSLambdaBasicExecutionRole")
echo "Policy attached."

# IAM roles can take a few seconds to propagate
echo "Waiting for IAM role to propagate..."
sleep 10
```

```
################################################################################
# Step 2: Create Lambda function
################################################################################

echo ""
echo "==========================================="
echo "Step 2: Create Lambda function"
echo "==========================================="
echo ""

echo "Creating deployment package..."
ORIGINAL_DIR=$(pwd)
cd "$TEMP_DIR" || exit 1
zip -j function.zip "$CODE_FILE" > /dev/null 2>&1 || {
    echo "ERROR: Failed to create deployment package"
    exit 1
}
cd "$ORIGINAL_DIR" || exit 1

if [ ! -f "${TEMP_DIR}/function.zip" ]; then
    echo "ERROR: Deployment package creation failed"
    exit 1
fi

echo "Creating Lambda function: ${FUNCTION_NAME}"
echo "  Runtime: ${RUNTIME}"
echo "  Handler: ${HANDLER}"
echo ""

CREATE_OUTPUT=$(aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime "$RUNTIME" \
    --role "$ROLE_ARN" \
    --handler "$HANDLER" \
    --architectures x86_64 \
    --zip-file "fileb://${TEMP_DIR}/function.zip" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted \
    --query '[FunctionName, FunctionArn, Runtime, State]' \
    --output text 2>&1)

if [ -z "$CREATE_OUTPUT" ]; then
    echo "ERROR: Failed to create Lambda function"
    exit 1
```

```
fi

echo "$CREATE_OUTPUT"
CREATED_RESOURCES+=("lambda-function:${FUNCTION_NAME}")

wait_for_resource "Lambda function to become Active" \
    "aws lambda get-function-configuration --function-name ${FUNCTION_NAME} --
query State --output text" \
    "Active"

################################################################################
# Step 3: Invoke the function
################################################################################

echo ""
echo "========================================="
echo "Step 3: Invoke the function"
echo "========================================="
echo ""

TEST_EVENT='{"length": 6, "width": 7}'
echo "Invoking function with test event: ${TEST_EVENT}"
echo ""

echo "$TEST_EVENT" > "${TEMP_DIR}/test-event.json"

if ! validate_input "$TEST_EVENT" '"length": [0-9]+, "width": [0-9]+'; then
    echo "ERROR: Invalid test event format"
    exit 1
fi

INVOKE_OUTPUT=$(aws lambda invoke \
    --function-name "$FUNCTION_NAME" \
    --payload "fileb://${TEMP_DIR}/test-event.json" \
    --cli-read-timeout 30 \
    "${TEMP_DIR}/response.json" 2>&1)
echo "$INVOKE_OUTPUT"

if [ ! -f "${TEMP_DIR}/response.json" ]; then
    echo "ERROR: No response file generated"
    exit 1
fi

RESPONSE=$(cat "${TEMP_DIR}/response.json")
```

```
echo ""
echo "Function response: ${RESPONSE}"
echo ""

if echo "$INVOKE_OUTPUT" | grep -qi "functionerror"; then
    echo "WARNING: Function returned an error."
fi

################################################################################
# Step 4: View CloudWatch logs
################################################################################

echo ""
echo "========================================="
echo "Step 4: View CloudWatch Logs"
echo "========================================="
echo ""

echo "Log group: ${LOG_GROUP_NAME}"
echo ""

echo "Waiting for CloudWatch logs to be available..."

LOG_STREAMS=""
for i in $(seq 1 6); do
    LOG_STREAMS=$(aws logs describe-log-streams \
        --log-group-name "$LOG_GROUP_NAME" \
        --order-by LastEventTime \
        --descending \
        --query 'logStreams[0].logStreamName' \
        --output text 2>/dev/null) || true
    if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
        break
    fi
    LOG_STREAMS=""
    sleep 5
done

if [ -n "$LOG_STREAMS" ] && [ "$LOG_STREAMS" != "None" ]; then
    echo "Latest log stream: ${LOG_STREAMS}"
    echo ""
    echo "--- Log events ---"
    LOG_EVENTS=$(aws logs get-log-events \
        --log-group-name "$LOG_GROUP_NAME" \
```

```
        --log-stream-name "$LOG_STREAMS" \
        --query 'events[].message' \
        --output text 2>&1) || true
    echo "$LOG_EVENTS"
    echo "--- End of log events ---"
else
    echo "No log streams found yet. Logs may take a moment to appear."
    echo "You can view them in the CloudWatch console:"
    echo "  Log group: ${LOG_GROUP_NAME}"
fi

CREATED_RESOURCES+=("log-group:${LOG_GROUP_NAME}")

aws logs tag-log-group \
    --log-group-name "$LOG_GROUP_NAME" \
    --tags project=doc-smith,tutorial=lambda-gettingstarted

################################################################################
# Summary and cleanup
################################################################################

echo ""
echo "==========================================="
echo "SUMMARY"
echo "==========================================="
echo ""
echo "Resources created:"
echo "  IAM role:          ${ROLE_NAME}"
echo "  Lambda function:   ${FUNCTION_NAME}"
echo "  CloudWatch logs:   ${LOG_GROUP_NAME}"
echo ""
echo "==========================================="
echo "CLEANUP"
echo "==========================================="
echo ""
echo "Cleaning up all created resources..."
cleanup_resources

echo ""
echo "Done."
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy

- CreateFunction
- CreateRole
- DeleteFunction
- DeleteLogGroup
- DeleteRole
- DescribeLogStreams
- DetachRolePolicy
- GetFunction
- GetFunctionConfiguration
- GetLogEvents
- Invocar

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de data warehouses com tecnologia sem servidor usando a CLI

O código de exemplo abaixo mostra como:

- Usar secrets-manager CreateSecret.
- Usar secrets-manager DeleteSecret.
- Usar secrets-manager GetSecretValue.
- Usar redshift CreateNamespace.
- Usar redshift CreateWorkgroup.
- Usar redshift DeleteNamespace.
- Usar iam CreateRole

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon Redshift Serverless Tutorial Script with Secrets Manager (No jq
 dependency)
# This script creates a Redshift Serverless environment, loads sample data, and
 runs queries
# Uses AWS Secrets Manager for secure password management without requiring jq

# Set up logging
LOG_FILE="redshift-serverless-tutorial-v4.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift Serverless tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
  local output=$1
  local cmd=$2

  if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Command failed: $cmd"
    echo "Output: $output"
    cleanup_resources
    exit 1
  fi
}

# Function to generate a secure password that meets Redshift requirements
generate_secure_password() {
  # Redshift password requirements:
  # - 8-64 characters
```

```
  # - At least one uppercase letter
  # - At least one lowercase letter
  # - At least one decimal digit
  # - Can contain printable ASCII characters except /, ", ', \, @, space

  local password=""
  local valid=false
  local attempts=0
  local max_attempts=10

  while [[ "$valid" == false && $attempts -lt $max_attempts ]]; do
    # Generate base password with safe characters
    local base=$(openssl rand -base64 12 | tr -d '/+=' | head -c 12)

    # Ensure we have at least one of each required character type
    local upper=$(echo "ABCDEFGHIJKLMNOPQRSTUVWXYZ" | fold -w1 | shuf -n1)
    local lower=$(echo "abcdefghijklmnopqrstuvwxyz" | fold -w1 | shuf -n1)
    local digit=$(echo "0123456789" | fold -w1 | shuf -n1)
    local special=$(echo "!#$%&*()_+-=[]{}|;:,.<>?" | fold -w1 | shuf -n1)

    # Combine and shuffle
    password="${base}${upper}${lower}${digit}${special}"
    password=$(echo "$password" | fold -w1 | shuf | tr -d '\n')

    # Validate password meets requirements
    if [[ ${#password} -ge 8 && ${#password} -le 64 ]] && \
       [[ "$password" =~ [A-Z] ]] && \
       [[ "$password" =~ [a-z] ]] && \
       [[ "$password" =~ [0-9] ]] && \
       [[ ! "$password" =~ [/\"\'\\@[:space:]] ]]; then
      valid=true
    fi

    ((attempts++))
  done

  if [[ "$valid" == false ]]; then
    echo "ERROR: Failed to generate valid password after $max_attempts attempts"
    exit 1
  fi

  echo "$password"
}
```

```
# Function to create secret in AWS Secrets Manager
create_secret() {
  local secret_name=$1
  local username=$2
  local password=$3
  local description=$4

  echo "Creating secret in AWS Secrets Manager: $secret_name"

  # Create the secret using AWS CLI without jq
  local secret_output=$(aws secretsmanager create-secret \
    --name "$secret_name" \
    --description "$description" \
    --secret-string "{\"username\":\"$username\",\"password\":\"$password\"}" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
 2>&1)

  if echo "$secret_output" | grep -i "error\|exception\|fail" > /dev/null; then
    echo "ERROR: Failed to create secret: $secret_output"
    return 1
  fi

  echo "Secret created successfully: $secret_name"
  return 0
}

# Function to retrieve password from AWS Secrets Manager
get_password_from_secret() {
  local secret_name=$1

  # Get the secret value and extract password using sed/grep instead of jq
  local secret_value=$(aws secretsmanager get-secret-value \
    --secret-id "$secret_name" \
    --query 'SecretString' \
    --output text 2>/dev/null)

  if [[ $? -eq 0 ]]; then
    # Extract password from JSON using sed
    echo "$secret_value" | sed -n 's/.*"password":"\([^"]*\)".*/\1/p'
  else
    echo ""
  fi
}
```

```
# Function to wait for a resource to be available
wait_for_resource() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be available..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>/dev/null)
    local status=$(echo "$output" | grep -o '"Status": "[^"]*' | cut -d'"' -f4 ||
 echo "")

    if [[ "$status" == "AVAILABLE" ]]; then
      echo "$resource_type $resource_name is now available"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name status:
 $status. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be
 available"
  return 1
}

# Function to wait for a resource to be deleted
wait_for_resource_deletion() {
  local resource_type=$1
  local resource_name=$2
  local max_attempts=$3
  local wait_seconds=$4
  local check_cmd=$5

  echo "Waiting for $resource_type $resource_name to be deleted..."

  for ((i=1; i<=$max_attempts; i++)); do
    local output=$($check_cmd 2>&1)

    if echo "$output" | grep -i "not found\|does not exist" > /dev/null; then
```

```
      echo "$resource_type $resource_name has been deleted"
      return 0
    fi

    echo "Attempt $i/$max_attempts: $resource_type $resource_name is still being
 deleted. Waiting $wait_seconds seconds..."
    sleep $wait_seconds
  done

  echo "ERROR: Timed out waiting for $resource_type $resource_name to be deleted"
  return 1
}

# Function to clean up resources
cleanup_resources() {
  echo ""
  echo "========================================="
  echo "CLEANUP CONFIRMATION"
  echo "========================================="
  echo "The following resources were created:"
  echo "- Redshift Serverless Workgroup: $WORKGROUP_NAME"
  echo "- Redshift Serverless Namespace: $NAMESPACE_NAME"
  echo "- IAM Role: $ROLE_NAME"
  echo "- Secrets Manager Secret: $SECRET_NAME"
  echo ""
  echo "Do you want to clean up all created resources? (y/n): "
  read -r CLEANUP_CHOICE

  if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the workgroup
    echo "Deleting Redshift Serverless workgroup $WORKGROUP_NAME..."
    WORKGROUP_DELETE_OUTPUT=$(aws redshift-serverless delete-workgroup --
workgroup-name "$WORKGROUP_NAME" 2>&1)
    echo "$WORKGROUP_DELETE_OUTPUT"

    # Wait for workgroup to be deleted before deleting namespace
    wait_for_resource_deletion "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-
serverless get-workgroup --workgroup-name $WORKGROUP_NAME"

    # Delete the namespace
    echo "Deleting Redshift Serverless namespace $NAMESPACE_NAME..."
```

```
    NAMESPACE_DELETE_OUTPUT=$(aws redshift-serverless delete-namespace --
namespace-name "$NAMESPACE_NAME" 2>&1)
    echo "$NAMESPACE_DELETE_OUTPUT"

    # Wait for namespace to be deleted
    wait_for_resource_deletion "namespace" "$NAMESPACE_NAME" 20 30 "aws redshift-
serverless get-namespace --namespace-name $NAMESPACE_NAME"

    # Delete the IAM role policy
    echo "Deleting IAM role policy..."
    POLICY_DELETE_OUTPUT=$(aws iam delete-role-policy --role-name "$ROLE_NAME" --
policy-name S3Access 2>&1)
    echo "$POLICY_DELETE_OUTPUT"

    # Delete the IAM role
    echo "Deleting IAM role $ROLE_NAME..."
    ROLE_DELETE_OUTPUT=$(aws iam delete-role --role-name "$ROLE_NAME" 2>&1)
    echo "$ROLE_DELETE_OUTPUT"

    # Delete the secret
    echo "Deleting Secrets Manager secret $SECRET_NAME..."
    SECRET_DELETE_OUTPUT=$(aws secretsmanager delete-secret --secret-id
 "$SECRET_NAME" --force-delete-without-recovery 2>&1)
    echo "$SECRET_DELETE_OUTPUT"

    echo "Cleanup completed."
  else
    echo "Cleanup skipped. Resources will remain in your AWS account."
  fi
}

# Check if required tools are available
if ! command -v openssl &> /dev/null; then
    echo "ERROR: openssl is required but not installed. Please install openssl to
 continue."
    exit 1
fi

# Generate unique names for resources
RANDOM_SUFFIX=$(cat /dev/urandom | tr -dc 'a-z0-9' | head -c 6)
NAMESPACE_NAME="rs-namespace-${RANDOM_SUFFIX}"
WORKGROUP_NAME="rs-workgroup-${RANDOM_SUFFIX}"
ROLE_NAME="RedshiftServerlessS3Role-${RANDOM_SUFFIX}"
SECRET_NAME="redshift-serverless-admin-${RANDOM_SUFFIX}"
```

```
DB_NAME="dev"
ADMIN_USERNAME="admin"

# Generate secure password
echo "Generating secure password..."
ADMIN_PASSWORD=$(generate_secure_password)

# Create secret in AWS Secrets Manager
create_secret "$SECRET_NAME" "$ADMIN_USERNAME" "$ADMIN_PASSWORD" "Admin
 credentials for Redshift Serverless namespace $NAMESPACE_NAME"
if [[ $? -ne 0 ]]; then
  echo "ERROR: Failed to create secret in AWS Secrets Manager"
  exit 1
fi

# Track created resources
CREATED_RESOURCES=()

echo "Using the following resource names:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- IAM Role: $ROLE_NAME"
echo "- Secret: $SECRET_NAME"
echo "- Database: $DB_NAME"
echo "- Admin Username: $ADMIN_USERNAME"
echo "- Admin Password: [STORED IN SECRETS MANAGER]"

# Step 1: Create IAM role for S3 access
echo "Creating IAM role for Redshift Serverless S3 access..."

# Create trust policy document
cat > redshift-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift-serverless.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
EOF

# Create S3 access policy document
cat > redshift-s3-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

# Create IAM role
echo "Creating IAM role $ROLE_NAME..."
ROLE_OUTPUT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://redshift-trust-policy.json 2>&1)
echo "$ROLE_OUTPUT"
check_error "$ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless
CREATED_RESOURCES+=("IAM Role: $ROLE_NAME")

# Attach S3 policy to the role
echo "Attaching S3 access policy to role $ROLE_NAME..."
POLICY_OUTPUT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 S3Access --policy-document file://redshift-s3-policy.json 2>&1)
echo "$POLICY_OUTPUT"
check_error "$POLICY_OUTPUT" "aws iam put-role-policy"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"
```

```
# Step 2: Create a namespace
echo "Creating Redshift Serverless namespace $NAMESPACE_NAME..."
NAMESPACE_OUTPUT=$(aws redshift-serverless create-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --admin-username "$ADMIN_USERNAME" \
  --admin-user-password "$ADMIN_PASSWORD" \
  --db-name "$DB_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$NAMESPACE_OUTPUT"
check_error "$NAMESPACE_OUTPUT" "aws redshift-serverless create-namespace"
CREATED_RESOURCES+=("Redshift Serverless Namespace: $NAMESPACE_NAME")

# Wait for namespace to be available
wait_for_resource "namespace" "$NAMESPACE_NAME" 10 30 "aws redshift-serverless
 get-namespace --namespace-name $NAMESPACE_NAME"

# Associate IAM role with namespace
echo "Associating IAM role with namespace..."
UPDATE_NAMESPACE_OUTPUT=$(aws redshift-serverless update-namespace \
  --namespace-name "$NAMESPACE_NAME" \
  --iam-roles "$ROLE_ARN" 2>&1)
echo "$UPDATE_NAMESPACE_OUTPUT"
check_error "$UPDATE_NAMESPACE_OUTPUT" "aws redshift-serverless update-namespace"

# Step 3: Create a workgroup
echo "Creating Redshift Serverless workgroup $WORKGROUP_NAME..."
WORKGROUP_OUTPUT=$(aws redshift-serverless create-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --namespace-name "$NAMESPACE_NAME" \
  --base-capacity 8 \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-serverless 2>&1)
echo "$WORKGROUP_OUTPUT"
check_error "$WORKGROUP_OUTPUT" "aws redshift-serverless create-workgroup"
CREATED_RESOURCES+=("Redshift Serverless Workgroup: $WORKGROUP_NAME")

# Wait for workgroup to be available
wait_for_resource "workgroup" "$WORKGROUP_NAME" 20 30 "aws redshift-serverless
 get-workgroup --workgroup-name $WORKGROUP_NAME"

# Get workgroup endpoint
WORKGROUP_ENDPOINT=$(aws redshift-serverless get-workgroup \
  --workgroup-name "$WORKGROUP_NAME" \
  --query 'workgroup.endpoint.address' \
  --output text)
```

```
echo "Workgroup endpoint: $WORKGROUP_ENDPOINT"

# Wait additional time for the endpoint to be fully operational
echo "Waiting for endpoint to be fully operational..."
sleep 60

# Step 4: Create tables for sample data
echo "Creating tables for sample data..."

# Create users table
echo "Creating users table..."
USERS_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE users(
    userid INTEGER NOT NULL DISTKEY SORTKEY,
    username CHAR(8),
    firstname VARCHAR(30),
    lastname VARCHAR(30),
    city VARCHAR(30),
    state CHAR(2),
    email VARCHAR(100),
    phone CHAR(14),
    likesports BOOLEAN,
    liketheatre BOOLEAN,
    likeconcerts BOOLEAN,
    likejazz BOOLEAN,
    likeclassical BOOLEAN,
    likeopera BOOLEAN,
    likerock BOOLEAN,
    likevegas BOOLEAN,
    likebroadway BOOLEAN,
    likemusicals BOOLEAN
  );" 2>&1)
echo "$USERS_TABLE_OUTPUT"
check_error "$USERS_TABLE_OUTPUT" "aws redshift-data execute-statement (users
 table)"
USERS_QUERY_ID=$(echo "$USERS_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for users table creation to complete..."
sleep 5
```

```
# Create event table
echo "Creating event table..."
EVENT_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE event(
    eventid INTEGER NOT NULL DISTKEY,
    venueid SMALLINT NOT NULL,
    catid SMALLINT NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    eventname VARCHAR(200),
    starttime TIMESTAMP
  );" 2>&1)
echo "$EVENT_TABLE_OUTPUT"
check_error "$EVENT_TABLE_OUTPUT" "aws redshift-data execute-statement (event
 table)"
EVENT_QUERY_ID=$(echo "$EVENT_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for query to complete
echo "Waiting for event table creation to complete..."
sleep 5

# Create sales table
echo "Creating sales table..."
SALES_TABLE_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "CREATE TABLE sales(
    salesid INTEGER NOT NULL,
    listid INTEGER NOT NULL DISTKEY,
    sellerid INTEGER NOT NULL,
    buyerid INTEGER NOT NULL,
    eventid INTEGER NOT NULL,
    dateid SMALLINT NOT NULL SORTKEY,
    qtysold SMALLINT NOT NULL,
    pricepaid DECIMAL(8,2),
    commission DECIMAL(8,2),
    saletime TIMESTAMP
  );" 2>&1)
echo "$SALES_TABLE_OUTPUT"
check_error "$SALES_TABLE_OUTPUT" "aws redshift-data execute-statement (sales
 table)"
```

```
SALES_QUERY_ID=$(echo "$SALES_TABLE_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"'
 -f4)

# Wait for tables to be created
echo "Waiting for tables to be created..."
sleep 10

# Step 5: Load sample data from Amazon S3
echo "Loading sample data from Amazon S3..."

# Load data into users table
echo "Loading data into users table..."
USERS_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY users
    FROM 's3://redshift-downloads/tickit/allusers_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$USERS_LOAD_OUTPUT"
check_error "$USERS_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 users)"
USERS_LOAD_QUERY_ID=$(echo "$USERS_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for users data loading to complete..."
sleep 10

# Load data into event table
echo "Loading data into event table..."
EVENT_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY event
    FROM 's3://redshift-downloads/tickit/allevents_pipe.txt'
    DELIMITER '|'
    TIMEFORMAT 'YYYY-MM-DD HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$EVENT_LOAD_OUTPUT"
```

```
check_error "$EVENT_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 event)"
EVENT_LOAD_QUERY_ID=$(echo "$EVENT_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for event data loading to complete..."
sleep 10

# Load data into sales table
echo "Loading data into sales table..."
SALES_LOAD_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "COPY sales
    FROM 's3://redshift-downloads/tickit/sales_tab.txt'
    DELIMITER '\t'
    TIMEFORMAT 'MM/DD/YYYY HH:MI:SS'
    IGNOREHEADER 1
    IAM_ROLE '$ROLE_ARN';" 2>&1)
echo "$SALES_LOAD_OUTPUT"
check_error "$SALES_LOAD_OUTPUT" "aws redshift-data execute-statement (load
 sales)"
SALES_LOAD_QUERY_ID=$(echo "$SALES_LOAD_OUTPUT" | grep -o '"Id": "[^"]*' | cut -
d'"' -f4)

# Wait for data loading to complete
echo "Waiting for sales data loading to complete..."
sleep 30

# Step 6: Run sample queries
echo "Running sample queries..."

# Query 1: Find top 10 buyers by quantity
echo "Running query: Find top 10 buyers by quantity..."
QUERY1_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT firstname, lastname, total_quantity
    FROM (SELECT buyerid, sum(qtysold) total_quantity
          FROM sales
          GROUP BY buyerid
          ORDER BY total_quantity desc limit 10) Q, users
    WHERE Q.buyerid = userid
```

```
    ORDER BY Q.total_quantity desc;" 2>&1)
echo "$QUERY1_OUTPUT"
check_error "$QUERY1_OUTPUT" "aws redshift-data execute-statement (query 1)"
QUERY1_ID=$(echo "$QUERY1_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 1 to complete..."
sleep 10

# Get query 1 results
echo "Getting results for query 1..."
QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
echo "$QUERY1_STATUS_OUTPUT"
check_error "$QUERY1_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 1)"

QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY1_STATUS" == "FINISHED" ]; then
  QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID" 2>&1)
  echo "Query 1 Results:"
  echo "$QUERY1_RESULTS"
else
  echo "Query 1 is not yet complete. Status: $QUERY1_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
  QUERY1_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY1_ID"
 2>&1)
  QUERY1_STATUS=$(echo "$QUERY1_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY1_STATUS" == "FINISHED" ]; then
    QUERY1_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY1_ID"
 2>&1)
    echo "Query 1 Results:"
    echo "$QUERY1_RESULTS"
  else
    echo "Query 1 is still not complete. Status: $QUERY1_STATUS"
  fi
fi
```

```
# Query 2: Find events in the 99.9 percentile in terms of all time total sales
echo "Running query: Find events in the 99.9 percentile in terms of all time
 total sales..."
QUERY2_OUTPUT=$(aws redshift-data execute-statement \
  --database "$DB_NAME" \
  --workgroup-name "$WORKGROUP_NAME" \
  --sql "SELECT eventname, total_price
    FROM (SELECT eventid, total_price, ntile(1000) over(order by total_price
 desc) as percentile
          FROM (SELECT eventid, sum(pricepaid) total_price
                FROM sales
                GROUP BY eventid)) Q, event E
    WHERE Q.eventid = E.eventid
    AND percentile = 1
    ORDER BY total_price desc;" 2>&1)
echo "$QUERY2_OUTPUT"
check_error "$QUERY2_OUTPUT" "aws redshift-data execute-statement (query 2)"
QUERY2_ID=$(echo "$QUERY2_OUTPUT" | grep -o '"Id": "[^"]*' | cut -d'"' -f4)

# Wait for query to complete
echo "Waiting for query 2 to complete..."
sleep 10

# Get query 2 results
echo "Getting results for query 2..."
QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
echo "$QUERY2_STATUS_OUTPUT"
check_error "$QUERY2_STATUS_OUTPUT" "aws redshift-data describe-statement (query
 2)"

QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut -
d'"' -f4)
if [ "$QUERY2_STATUS" == "FINISHED" ]; then
  QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID" 2>&1)
  echo "Query 2 Results:"
  echo "$QUERY2_RESULTS"
else
  echo "Query 2 is not yet complete. Status: $QUERY2_STATUS"
  echo "Waiting additional time for query to complete..."
  sleep 20

  # Check again
```

```
  QUERY2_STATUS_OUTPUT=$(aws redshift-data describe-statement --id "$QUERY2_ID"
 2>&1)
  QUERY2_STATUS=$(echo "$QUERY2_STATUS_OUTPUT" | grep -o '"Status": "[^"]*' | cut
 -d'"' -f4)

  if [ "$QUERY2_STATUS" == "FINISHED" ]; then
    QUERY2_RESULTS=$(aws redshift-data get-statement-result --id "$QUERY2_ID"
 2>&1)
    echo "Query 2 Results:"
    echo "$QUERY2_RESULTS"
  else
    echo "Query 2 is still not complete. Status: $QUERY2_STATUS"
  fi
fi

# Summary
echo ""
echo "========================================="
echo "TUTORIAL SUMMARY"
echo "========================================="
echo "You have successfully:"
echo "1. Created a Redshift Serverless namespace and workgroup"
echo "2. Created an IAM role with S3 access permissions"
echo "3. Stored admin credentials securely in AWS Secrets Manager"
echo "4. Created tables for sample data"
echo "5. Loaded sample data from Amazon S3"
echo "6. Run sample queries on the data"
echo ""
echo "Redshift Serverless Resources:"
echo "- Namespace: $NAMESPACE_NAME"
echo "- Workgroup: $WORKGROUP_NAME"
echo "- Database: $DB_NAME"
echo "- Endpoint: $WORKGROUP_ENDPOINT"
echo "- Credentials Secret: $SECRET_NAME"
echo ""
echo "To connect to your Redshift Serverless database using SQL tools:"
echo "- Host: $WORKGROUP_ENDPOINT"
echo "- Database: $DB_NAME"
echo "- Username: $ADMIN_USERNAME"
echo "- Password: Retrieve from AWS Secrets Manager secret '$SECRET_NAME'"
echo ""
echo "To retrieve the password from Secrets Manager (without jq):"
echo "aws secretsmanager get-secret-value --secret-id $SECRET_NAME --query
 'SecretString' --output text | sed -n 's/.*\"password\":\"\([^\"]*\)\".*/\1/p'"
```

```
echo ""

# Clean up temporary files
rm -f redshift-trust-policy.json redshift-s3-policy.json

# Clean up resources
cleanup_resources

echo "Tutorial completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - CreateNamespace
  - CreateRole
  - CreateSecret
  - CreateWorkgroup
  - DeleteNamespace
  - DeleteRole
  - DeleteRolePolicy
  - DeleteSecret
  - DeleteWorkgroup
  - GetNamespace
  - GetRole
  - GetSecretValue
  - GetWorkgroup
  - PutRolePolicy
  - UpdateNamespace

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos do gerenciamento de configurações

O código de exemplo abaixo mostra como:

- Criar um bucket do Amazon S3.
- Criar um tópico do Amazon SNS.
- Criar um perfil do IAM para o Config.
- Configurar o gravador de configuração do Config.
- Configurar o canal de entrega do Config.
- Iniciar o gravador de configuração.
- Verificar a configuração do Config

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
```

```
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi

    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
```

```
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}

# Function to display created resources
display_resources() {
    echo ""
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
```

```
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "============================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
```

```
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"

# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"

echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
```

```
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
```

```
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"
```

```
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"

echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "============================================"
echo "CLEANUP CONFIRMATION"
```

```
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel
  - DeleteRole
  - DeleteRolePolicy
  - DeleteTopic
  - DescribeConfigurationRecorderStatus
  - DescribeConfigurationRecorders
  - DescribeDeliveryChannels
  - DetachRolePolicy
  - GetCallerIdentity
  - PutConfigurationRecorder
  - PutDeliveryChannel
  - PutPublicAccessBlock
  - PutRolePolicy

- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos da proteção de dispositivos da Internet das Coisas

O código de exemplo abaixo mostra como:

- Criar perfis do IAM obrigatórios.
- Habiltiar verificações de auditoria do IoT Device Defender.
- Executar uma auditoria sob demanda
- Criar uma ação de mitigação.
- Aplicar ações de mitigação às descobertas.
- Configurar notificações do SNS (opcional).
- Habilitar o registro em log de IoT

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS IoT Device Defender Getting Started Script
# This script demonstrates how to use AWS IoT Device Defender to enable audit
 checks,
# view audit results, create mitigation actions, and apply them to findings.
```

```
set -euo pipefail

# Set up logging
LOG_FILE="iot-device-defender-script-$(date +%Y%m%d%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "AWS IoT Device Defender Getting Started Script"
echo "================================================="
echo "Starting script execution at $(date)"
echo ""

# Function to check for errors in command output
check_error() {
    if echo "$1" | grep -iE "An error occurred|Exception|Failed|usage: aws" > /
dev/null; then
        echo "ERROR: Command failed with the following output:"
        echo "$1"
        return 1
    fi
    return 0
}

# Function to safely extract JSON values using jq
extract_json_value() {
    local json="$1"
    local key="$2"
    echo "$json" | jq -r ".${key} // empty" 2>/dev/null || echo ""
}

# Function to validate JSON
validate_json() {
    local json="$1"
    echo "$json" | jq empty 2>/dev/null
}

# Function to check AWS CLI availability
check_aws_cli() {
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed or not in PATH"
        return 1
    fi
    if ! command -v jq &> /dev/null; then
        echo "ERROR: jq is not installed or not in PATH"
```

```
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    local account_id
    account_id=$(aws sts get-caller-identity --query 'Account' --output text 2>/
dev/null) || true
    if [ -z "$account_id" ]; then
        echo "ERROR: Could not retrieve AWS account ID"
        return 1
    fi
    echo "$account_id"
    return 0
}

# Function to create IAM roles with retry logic
create_iam_role() {
    local ROLE_NAME=$1
    local TRUST_POLICY=$2
    local MANAGED_POLICY=$3
    local RETRY_COUNT=0
    local MAX_RETRIES=3

    echo "Creating IAM role: $ROLE_NAME"

    # Validate trust policy JSON
    if ! validate_json "$TRUST_POLICY"; then
        echo "ERROR: Invalid trust policy JSON for role $ROLE_NAME"
        return 1
    fi

    # Check if role already exists
    if aws iam get-role --role-name "$ROLE_NAME" >/dev/null 2>&1; then
        echo "Role $ROLE_NAME already exists, skipping creation"
        ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn'
 --output text 2>/dev/null) || true
        if [ -z "$ROLE_ARN" ]; then
            echo "ERROR: Could not retrieve ARN for existing role $ROLE_NAME"
            return 1
        fi
        echo "Role ARN: $ROLE_ARN"
```

```
        return 0
    fi

    # Create the role with trust policy and retry logic
    while [ $RETRY_COUNT -lt $MAX_RETRIES ]; do
        ROLE_RESULT=$(aws iam create-role \
            --role-name "$ROLE_NAME" \
            --assume-role-policy-document "$TRUST_POLICY" 2>&1) || true

        if check_error "$ROLE_RESULT"; then
            break
        fi

        RETRY_COUNT=$((RETRY_COUNT + 1))
        if [ $RETRY_COUNT -lt $MAX_RETRIES ]; then
            echo "Retrying role creation (attempt $((RETRY_COUNT + 1))/
$MAX_RETRIES)..."
            sleep $((RETRY_COUNT * 2))
        fi
    done

    if ! check_error "$ROLE_RESULT"; then
        echo "Failed to create role $ROLE_NAME after $MAX_RETRIES attempts"
        return 1
    fi

    aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

    # For IoT logging role, create an inline policy instead of using a managed
 policy
    if [[ "$ROLE_NAME" == "AWSIoTLoggingRole" ]]; then
        local LOGGING_POLICY
        LOGGING_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:PutMetricFilter",
```

```
                "logs:PutRetentionPolicy",
                "logs:GetLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        }
    ]
}
EOF
)

        if ! validate_json "$LOGGING_POLICY"; then
            echo "ERROR: Invalid logging policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$LOGGING_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
            return 1
        fi
    elif [[ "$ROLE_NAME" == "IoTMitigationActionErrorLoggingRole" ]]; then
        local MITIGATION_POLICY
        MITIGATION_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:UpdateCACertificate",
                "iot:UpdateCertificate",
                "iot:SetV2LoggingOptions",
                "iot:SetLoggingOptions",
                "iot:AddThingToThingGroup"
            ],
            "Resource": "arn:aws:iot:*:*:*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "iot.amazonaws.com"
                }
            }
        }
    ]
}
EOF
)

        if ! validate_json "$MITIGATION_POLICY"; then
            echo "ERROR: Invalid mitigation policy JSON"
            return 1
        fi

        POLICY_RESULT=$(aws iam put-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-name "${ROLE_NAME}Policy" \
            --policy-document "$MITIGATION_POLICY" 2>&1) || true

        if ! check_error "$POLICY_RESULT"; then
            echo "Failed to attach inline policy to role $ROLE_NAME"
            return 1
        fi
    else
        # Attach managed policy to role if provided
        if [ -n "$MANAGED_POLICY" ]; then
            ATTACH_RESULT=$(aws iam attach-role-policy \
                --role-name "$ROLE_NAME" \
                --policy-arn "$MANAGED_POLICY" 2>&1) || true

            if ! check_error "$ATTACH_RESULT"; then
                echo "Failed to attach policy to role $ROLE_NAME"
                return 1
            fi
        fi
    fi

    echo "Role $ROLE_NAME created successfully"

    # Get the role ARN with error handling
```

```
    ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --
output text 2>/dev/null) || true
    if [ -z "$ROLE_ARN" ]; then
        echo "ERROR: Could not retrieve ARN for newly created role $ROLE_NAME"
        return 1
    fi
    echo "Role ARN: $ROLE_ARN"
    return 0
}

# Array to store created resources for cleanup
declare -a CREATED_RESOURCES

# Validate prerequisites
echo "Validating prerequisites..."
if ! check_aws_cli; then
    echo "ERROR: Prerequisites not met"
    exit 1
fi

ACCOUNT_ID=$(get_account_id) || exit 1
echo "AWS Account ID: $ACCOUNT_ID"
echo ""

# Step 1: Create IAM roles needed for the tutorial
echo "=================================================="
echo "Step 1: Creating required IAM roles"
echo "=================================================="

# Create IoT Device Defender Audit role
IOT_DEFENDER_AUDIT_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

```
)

if ! create_iam_role "AWSIoTDeviceDefenderAuditRole"
 "$IOT_DEFENDER_AUDIT_TRUST_POLICY" "arn:aws:iam::aws:policy/service-role/
AWSIoTDeviceDefenderAudit"; then
    echo "ERROR: Failed to create audit role"
    exit 1
fi
AUDIT_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTDeviceDefenderAuditRole")

# Create IoT Logging role
IOT_LOGGING_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "AWSIoTLoggingRole" "$IOT_LOGGING_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create logging role"
    exit 1
fi
LOGGING_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: AWSIoTLoggingRole")

# Create IoT Mitigation Action role
IOT_MITIGATION_TRUST_POLICY=$(cat <<'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
)

if ! create_iam_role "IoTMitigationActionErrorLoggingRole"
 "$IOT_MITIGATION_TRUST_POLICY" ""; then
    echo "ERROR: Failed to create mitigation role"
    exit 1
fi
MITIGATION_ROLE_ARN=$ROLE_ARN
CREATED_RESOURCES+=("IAM Role: IoTMitigationActionErrorLoggingRole")

# Wait for IAM role propagation
echo "Waiting for IAM role propagation..."
sleep 5

# Step 2: Enable audit checks
echo ""
echo "================================================="
echo "Step 2: Enabling AWS IoT Device Defender audit checks"
echo "================================================="

# Get current audit configuration
echo "Getting current audit configuration..."
CURRENT_CONFIG=$(aws iot describe-account-audit-configuration --output json 2>&1)
 || true
if validate_json "$CURRENT_CONFIG"; then
    echo "$CURRENT_CONFIG" | jq '.' 2>/dev/null || echo "Could not parse current
 configuration"
fi

# Enable specific audit checks with proper JSON escaping
echo "Enabling audit checks..."
AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":true}}'

if ! validate_json "$AUDIT_CONFIG"; then
    echo "ERROR: Invalid audit configuration JSON"
    exit 1
fi
```

```
UPDATE_RESULT=$(aws iot update-account-audit-configuration \
  --role-arn "$AUDIT_ROLE_ARN" \
  --audit-check-configurations "$AUDIT_CONFIG" 2>&1) || true

if ! check_error "$UPDATE_RESULT"; then
    echo "Failed to update audit configuration"
    exit 1
fi

echo "Audit checks enabled successfully"

# Step 3: Run an on-demand audit
echo ""
echo "=================================================="
echo "Step 3: Running an on-demand audit"
echo "=================================================="

echo "Starting on-demand audit task..."
AUDIT_TASK_RESULT=$(aws iot start-on-demand-audit-task \
  --target-check-names LOGGING_DISABLED_CHECK --output json 2>&1) || true

if ! check_error "$AUDIT_TASK_RESULT"; then
    echo "Failed to start on-demand audit task"
    exit 1
fi

TASK_ID=$(extract_json_value "$AUDIT_TASK_RESULT" "taskId")
if [ -z "$TASK_ID" ]; then
    echo "ERROR: Could not extract task ID from response"
    exit 1
fi

echo "Audit task started with ID: $TASK_ID"
CREATED_RESOURCES+=("Audit Task: $TASK_ID")

# Tag the audit task via IoT service
aws iot tag-resource --resource-arn "arn:aws:iot:$(aws configure get region):
${ACCOUNT_ID}:audittask/${TASK_ID}" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-device-defender-gs 2>&1 || true

# Wait for the audit task to complete
echo "Waiting for audit task to complete (this may take a few minutes)..."
TASK_STATUS="IN_PROGRESS"
TIMEOUT=0
```

```
MAX_TIMEOUT=600
POLL_INTERVAL=15

while [ "$TASK_STATUS" != "COMPLETED" ]; do
    if [ $TIMEOUT -ge $MAX_TIMEOUT ]; then
        echo "WARNING: Audit task did not complete within ${MAX_TIMEOUT} seconds,
 continuing..."
        break
    fi

    sleep "$POLL_INTERVAL"
    TIMEOUT=$((TIMEOUT + POLL_INTERVAL))

    TASK_DETAILS=$(aws iot describe-audit-task --task-id "$TASK_ID" --output json
 2>&1) || true
    if validate_json "$TASK_DETAILS"; then
        TASK_STATUS=$(extract_json_value "$TASK_DETAILS" "taskStatus")
        echo "Current task status: $TASK_STATUS (elapsed: ${TIMEOUT}s)"

        if [ "$TASK_STATUS" = "FAILED" ]; then
            echo "WARNING: Audit task failed, continuing with script..."
            FAILURE_REASON=$(extract_json_value "$TASK_DETAILS"
 "taskStatistics.failedChecksNotApplicable")
            if [ -n "$FAILURE_REASON" ]; then
                echo "Reason: $FAILURE_REASON"
            fi
            break
        fi
    else
        echo "WARNING: Could not parse task details, retrying..."
    fi
done

echo "Audit task processing completed"

# Get audit findings (non-blocking)
echo "Getting audit findings..."
FINDINGS=$(aws iot list-audit-findings \
  --task-id "$TASK_ID" --output json 2>&1) || true

if validate_json "$FINDINGS"; then
    FINDING_COUNT=$(echo "$FINDINGS" | jq '.findings | length' 2>/dev/null ||
 echo "0")
    echo "Audit findings count: $FINDING_COUNT"
```

```
    if [ "$FINDING_COUNT" -gt 0 ]; then
        echo "Sample finding:"
        echo "$FINDINGS" | jq '.findings[0]' 2>/dev/null || echo "Could not parse
 finding"
    fi
else
    echo "WARNING: Could not parse audit findings response"
    FINDINGS='{"findings":[]}'
fi

# Check if we have any non-compliant findings
FINDING_ID=$(extract_json_value "$FINDINGS" "findings[0].findingId")
if [ -n "$FINDING_ID" ]; then
    echo "Found non-compliant finding with ID: $FINDING_ID"
    HAS_FINDINGS=true
else
    echo "No non-compliant findings detected"
    HAS_FINDINGS=false
fi

# Step 4: Create a mitigation action
echo ""
echo "================================================="
echo "Step 4: Creating a mitigation action"
echo "================================================="

# Check if mitigation action already exists and delete it
if aws iot describe-mitigation-action --action-name "EnableErrorLoggingAction" >/
dev/null 2>&1; then
    echo "Mitigation action 'EnableErrorLoggingAction' already exists, deleting
 it first..."
    aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction"
 2>&1 || true
    sleep 2
fi

echo "Creating mitigation action to enable AWS IoT logging..."

# Build mitigation action parameters JSON
MITIGATION_PARAMS=$(cat <<EOF
{
  "enableIoTLoggingParams": {
    "roleArnForLogging": "$LOGGING_ROLE_ARN",
    "logLevel": "ERROR"
```

```
  }
}
EOF
)

if ! validate_json "$MITIGATION_PARAMS"; then
    echo "ERROR: Invalid mitigation parameters JSON"
    exit 1
fi

MITIGATION_RESULT=$(aws iot create-mitigation-action \
  --action-name "EnableErrorLoggingAction" \
  --role-arn "$MITIGATION_ROLE_ARN" \
  --action-params "$MITIGATION_PARAMS" --output json 2>&1) || true

if ! check_error "$MITIGATION_RESULT"; then
    echo "Failed to create mitigation action"
    exit 1
fi

if validate_json "$MITIGATION_RESULT"; then
    echo "Mitigation action created successfully"
    MITIGATION_ACTION_ARN=$(extract_json_value "$MITIGATION_RESULT" "actionArn")
    if [ -n "$MITIGATION_ACTION_ARN" ]; then
        echo "Mitigation Action ARN: $MITIGATION_ACTION_ARN"
        aws iot tag-resource --resource-arn "$MITIGATION_ACTION_ARN" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs 2>&1
 || true
    fi
else
    echo "WARNING: Could not validate mitigation action response, but action may
 have been created"
fi

CREATED_RESOURCES+=("Mitigation Action: EnableErrorLoggingAction")

# Step 5: Apply mitigation action to findings (if any)
if [ "$HAS_FINDINGS" = true ]; then
    echo ""
    echo "================================================="
    echo "Step 5: Applying mitigation action to findings"
    echo "================================================="

    MITIGATION_TASK_ID="MitigationTask-$(date +%s)"
```

```
    echo "Starting mitigation actions task with ID: $MITIGATION_TASK_ID"

    # Build target JSON
    TARGET_JSON=$(cat <<EOF
{
  "findingIds": ["$FINDING_ID"]
}
EOF
)

    if ! validate_json "$TARGET_JSON"; then
        echo "ERROR: Invalid target JSON"
        exit 1
    fi

    # Build audit check to actions mapping JSON
    AUDIT_CHECK_MAPPING=$(cat <<EOF
{
  "LOGGING_DISABLED_CHECK": ["EnableErrorLoggingAction"]
}
EOF
)

    if ! validate_json "$AUDIT_CHECK_MAPPING"; then
        echo "ERROR: Invalid audit check mapping JSON"
        exit 1
    fi

    MITIGATION_TASK_RESULT=$(aws iot start-audit-mitigation-actions-task \
      --task-id "$MITIGATION_TASK_ID" \
      --target "$TARGET_JSON" \
      --audit-check-to-actions-mapping "$AUDIT_CHECK_MAPPING" --output json 2>&1)
 || true

    if ! check_error "$MITIGATION_TASK_RESULT"; then
        echo "WARNING: Failed to start mitigation actions task, continuing..."
    else
        echo "Mitigation actions task started successfully"
        CREATED_RESOURCES+=("Mitigation Task: $MITIGATION_TASK_ID")
    fi
else
    echo ""
    echo "================================================="
    echo "Step 5: Skipping mitigation action application (no findings)"
```

```
    echo "======================================================"
fi

# Step 6: Set up SNS notifications (optional)
echo ""
echo "======================================================"
echo "Step 6: Setting up SNS notifications"
echo "======================================================"

# Check if SNS topic already exists
SNS_TOPICS=$(aws sns list-topics --output json 2>&1) || true
TOPIC_ARN=""
if validate_json "$SNS_TOPICS"; then
    TOPIC_ARN=$(echo "$SNS_TOPICS" | jq -r '.Topics[] | select(.TopicArn |
 contains("IoTDDNotifications")) | .TopicArn' 2>/dev/null | head -1 || echo "")
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "SNS topic 'IoTDDNotifications' already exists, using existing topic..."
    echo "Topic ARN: $TOPIC_ARN"
else
    echo "Creating SNS topic for notifications..."
    SNS_RESULT=$(aws sns create-topic --name "IoTDDNotifications" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-iot-device-defender-gs --
output json 2>&1) || true

    if ! check_error "$SNS_RESULT"; then
        echo "WARNING: Failed to create SNS topic, continuing..."
        SNS_RESULT=""
    else
        TOPIC_ARN=$(extract_json_value "$SNS_RESULT" "TopicArn")
        if [ -n "$TOPIC_ARN" ]; then
            echo "SNS topic created with ARN: $TOPIC_ARN"
            CREATED_RESOURCES+=("SNS Topic: IoTDDNotifications")
        fi
    fi
fi

if [ -n "$TOPIC_ARN" ]; then
    echo "Updating audit configuration to enable SNS notifications..."

    # Build SNS notification configuration JSON
    SNS_CONFIG=$(cat <<EOF
{
```

```
  "SNS": {
    "targetArn": "$TOPIC_ARN",
    "roleArn": "$AUDIT_ROLE_ARN",
    "enabled": true
  }
}
EOF
)

    if ! validate_json "$SNS_CONFIG"; then
        echo "ERROR: Invalid SNS configuration JSON"
        exit 1
    fi

    SNS_UPDATE_RESULT=$(aws iot update-account-audit-configuration \
      --audit-notification-target-configurations "$SNS_CONFIG" 2>&1) || true

    if ! check_error "$SNS_UPDATE_RESULT"; then
        echo "WARNING: Failed to update audit configuration for SNS
 notifications"
    else
        echo "SNS notifications enabled successfully"
    fi
else
    echo "Skipping SNS configuration due to topic creation failure"
fi

# Step 7: Enable AWS IoT logging
echo ""
echo "================================================="
echo "Step 7: Enabling AWS IoT logging"
echo "================================================="

echo "Setting up AWS IoT logging options..."

LOGGING_RESULT=$(aws iot set-v2-logging-options \
  --role-arn "$LOGGING_ROLE_ARN" \
  --default-log-level "ERROR" 2>&1) || true

if ! check_error "$LOGGING_RESULT"; then
    echo "V2 logging setup failed, trying v1 logging..."

    V1_LOGGING_CONFIG=$(cat <<EOF
{
```

```
  "roleArn": "$LOGGING_ROLE_ARN",
  "logLevel": "ERROR"
}
EOF
)

    if ! validate_json "$V1_LOGGING_CONFIG"; then
        echo "ERROR: Invalid v1 logging configuration JSON"
        exit 1
    fi

    LOGGING_RESULT_V1=$(aws iot set-logging-options \
      --logging-options-payload "$V1_LOGGING_CONFIG" 2>&1) || true

    if ! check_error "$LOGGING_RESULT_V1"; then
        echo "WARNING: Failed to set up AWS IoT logging with both v1 and v2
 methods, continuing..."
    else
        echo "AWS IoT v1 logging enabled successfully"
    fi
else
    echo "AWS IoT v2 logging enabled successfully"
fi

# Verify logging is enabled
echo "Verifying logging configuration..."
LOGGING_CONFIG=$(aws iot get-v2-logging-options --output json 2>&1) || true
if [ -n "$LOGGING_CONFIG" ] && ! check_error "$LOGGING_CONFIG" && validate_json
 "$LOGGING_CONFIG"; then
    echo "Logging configuration verified"
    echo "$LOGGING_CONFIG" | jq '.' 2>/dev/null || echo "Configuration retrieved
 but could not display details"
else
    echo "Could not verify logging configuration, but setup completed"
fi

# Script completed successfully
echo ""
echo "================================================="
echo "AWS IoT Device Defender setup completed successfully!"
echo "================================================="
echo "The following resources were created:"
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "- $resource"
```

```
done
echo ""

# Cleanup phase
echo "========================================="
echo "CLEANUP"
echo "========================================="
echo "Starting automatic cleanup of resources..."
echo "Waiting 10 seconds before cleanup to allow resource stabilization..."
sleep 10

# Disable AWS IoT logging
echo "Disabling AWS IoT logging..."

DISABLE_V2_RESULT=$(aws iot set-v2-logging-options \
  --default-log-level "DISABLED" 2>&1) || true

if check_error "$DISABLE_V2_RESULT"; then
    echo "V2 logging disabled successfully"
else
    echo "Attempting v1 logging disable..."

    V1_DISABLE_CONFIG=$(cat <<'EOF'
{
  "logLevel": "DISABLED"
}
EOF
)

    DISABLE_V1_RESULT=$(aws iot set-logging-options \
      --logging-options-payload "$V1_DISABLE_CONFIG" 2>&1) || true

    if check_error "$DISABLE_V1_RESULT"; then
        echo "V1 logging disabled successfully"
    else
        echo "WARNING: Could not disable logging"
    fi
fi

# Delete mitigation action
echo "Deleting mitigation action..."
aws iot delete-mitigation-action --action-name "EnableErrorLoggingAction" 2>&1 ||
 true
```

```
# Reset audit configuration
echo "Resetting IoT Device Defender audit configuration..."
RESET_AUDIT_CONFIG='{"LOGGING_DISABLED_CHECK":{"enabled":false}}'
aws iot update-account-audit-configuration \
  --audit-check-configurations "$RESET_AUDIT_CONFIG" 2>&1 || true

# Delete SNS topic
echo "Deleting SNS topic..."
if [ -n "${TOPIC_ARN:-}" ] && [ "$TOPIC_ARN" != "null" ]; then
    aws sns delete-topic --topic-arn "$TOPIC_ARN" 2>&1 || true
fi

# Clean up IAM roles with improved error handling
echo "Cleaning up IAM roles..."

cleanup_role() {
    local role_name=$1
    echo "Cleaning up role: $role_name"

    if aws iam get-role --role-name "$role_name" >/dev/null 2>&1; then
        ROLE_POLICIES=$(aws iam list-role-policies --role-name "$role_name" --
output json 2>&1 || echo '{"PolicyNames":[]}')
        if validate_json "$ROLE_POLICIES"; then
            while IFS= read -r policy_name; do
                if [ -n "$policy_name" ] && [ "$policy_name" != "null" ]; then
                    echo "  Deleting inline policy: $policy_name"
                    aws iam delete-role-policy \
                        --role-name "$role_name" \
                        --policy-name "$policy_name" 2>&1 || true
                fi
            done < <(echo "$ROLE_POLICIES" | jq -r '.PolicyNames[]' 2>/dev/null
 || echo "")
        fi

        ATTACHED_POLICIES=$(aws iam list-attached-role-policies --role-name
 "$role_name" --output json 2>&1 || echo '{"AttachedPolicies":[]}')
        if validate_json "$ATTACHED_POLICIES"; then
            while IFS= read -r policy_arn; do
                if [ -n "$policy_arn" ] && [ "$policy_arn" != "null" ]; then
                    echo "  Detaching managed policy: $policy_arn"
                    aws iam detach-role-policy \
                        --role-name "$role_name" \
                        --policy-arn "$policy_arn" 2>&1 || true
                fi
```

```
            done < <(echo "$ATTACHED_POLICIES" | jq -r
 '.AttachedPolicies[].PolicyArn' 2>/dev/null || echo "")
        fi

        echo "  Deleting role: $role_name"
        aws iam delete-role --role-name "$role_name" 2>&1 || true
    else
        echo "  Role $role_name does not exist or already deleted"
    fi
}

cleanup_role "AWSIoTDeviceDefenderAuditRole"
cleanup_role "AWSIoTLoggingRole"
cleanup_role "IoTMitigationActionErrorLoggingRole"

echo "Cleanup completed successfully"

echo ""
echo "Script execution completed at $(date)"
echo "Log file: $LOG_FILE"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateMitigationAction
  - CreateRole
  - CreateTopic
  - DeleteAccountAuditConfiguration
  - DeleteMitigationAction
  - DeleteRole
  - DeleteRolePolicy
  - DeleteTopic
  - DescribeAccountAuditConfiguration
  - DescribeAuditTask
  - DetachRolePolicy
  - GetLoggingOptions
  - GetRole
  - GetV2LoggingOptions

- ListAuditFindings
- ListAuditMitigationActionsTasks
- ListMitigationActions
- ListRolePolicies
- ListTopics
- PutRolePolicy
- SetLoggingOptions
- SetV2LoggingOptions
- StartAuditMitigationActionsTask
- StartOnDemandAuditTask
- UpdateAccountAuditConfiguration

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de arquivos de atributos de machine learning

O código de exemplo abaixo mostra como:

- Configurar permissões do IAM.
- Criar um perfil de execução do SageMaker.
- Criar grupo de atributos.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
```

```
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
```

```
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=sagemaker-featurestore 2>&1
```

```
    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"
```

```
    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
```

```
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
```

```
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")
```

```
# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."
```

```
# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
```

```
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"

ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
```

```
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi
```

```
echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "============================================"
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "============================================"
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "============================================"
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "============================================"
    echo "CLEANUP CONFIRMATION"
    echo "============================================"
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
```

```
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateBucket
  - CreateFeatureGroup
  - CreateRole
  - DeleteBucket
  - DeleteFeatureGroup
  - DeleteRole
  - DescribeFeatureGroup
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListFeatureGroups
  - PutPublicAccessBlock
  - Rm

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de clusters gerenciados de kubernetes

O código de exemplo abaixo mostra como:

- Criar uma VPC para um cluster do EKS.
- Criar perfis do IAM para um cluster do EKS.
- Criação do seu cluster de EKS
- Configurar o kubectl para se comunicar com o cluster.
- Crie um grupo de nós gerenciados
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
```

```
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
```

```
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
```

```
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi
```

```
    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."

# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

```
# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
```

```
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"

# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
```

```
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"

NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
```

```
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
```

```
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
```

```
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup
  - DeleteRole
  - DeleteStack
  - DescribeCluster
  - DescribeNodegroup
  - DescribeStacks
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNodegroups
  - UpdateKubeconfig
  - Espere (CloudFormation)
  - Aguardar (Amazon EKS)

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de streaming gerenciado

O código de exemplo abaixo mostra como:

- Criar um cluster do Amazon MSK.
- Criar permissões do IAM para acessar o MSK.
- Criar uma máquina cliente
- Obter os agentes de bootstrap.
- Configurar a máquina cliente.
- Criar um tópico e produzir/consumir dados.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
```

```
echo "============================================="

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
```

```
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
}

# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi

    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)
```

```
                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"

                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
```

```
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete EC2 instance if it exists
    if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
        echo "Terminating EC2 instance: $INSTANCE_ID"
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
 to terminate instance"
        echo "Waiting for instance to terminate..."
        aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
 "Failed to wait for instance termination"
    fi

    # Delete MSK cluster first (to remove dependencies on security group)
    if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
        echo "Deleting MSK cluster: $CLUSTER_ARN"
        aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
 delete cluster"

        # Wait a bit for the cluster deletion to start
        echo "Waiting 30 seconds for cluster deletion to begin..."
        sleep 30
    fi

    # Remove security group references before attempting deletion
    if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
 "$CLIENT_SG_ID"; then
        remove_security_group_references "$CLIENT_SG_ID"

        echo "Deleting security group: $CLIENT_SG_ID"
        # Try multiple times with longer delays to ensure dependencies are
 removed
        for i in {1..10}; do
            if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
                echo "Security group deleted successfully"
                break
            fi
```

```
            echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
            sleep 30
        done
    fi

    # Delete key pair if it exists
    if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
        echo "Deleting key pair: $KEY_NAME"
        aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
    fi

    # Remove role from instance profile
    if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
        echo "Removing role from instance profile"
        aws iam remove-role-from-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" \
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
"$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
"$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
```

```
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
 role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
 delete policy"
    fi

    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")

    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)
```

```
            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
        done
    done

    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
```

```
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
```

```
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
```

```
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi

if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
```

```
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"
```

```
# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi
```

```
echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi

echo "Role added to instance profile"

# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="

# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"
```

```
# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi

echo "MSK security group ID: $MSK_SG_ID"

# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)
```

```
if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
```

```
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
```

```
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)

if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi

echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
```

```
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
echo "Getting bootstrap brokers"
MAX_RETRIES=10
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
```

```
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
done

echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
```

```
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
cd $KAFKA_ROOT/libs
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
```

```
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
```

```
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
echo "============================================="

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCE SUMMARY"
echo "============================================="
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "============================================="
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
```

```
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
```

```
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
echo ""
echo "=========================================="
echo "CLEANUP CONFIRMATION"
echo "=========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress

- CreateCluster
- CreateInstanceProfile
- CreateKeyPair
- CreatePolicy
- CreateRole
- CreateSecurityGroup
- DeleteCluster
- DeleteInstanceProfile
- DeleteKeyPair
- DeletePolicy
- DeleteRole
- DeleteSecurityGroup
- DescribeAvailabilityZones
- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress

- RunInstances
- TerminateInstances
- Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de clusters de data warehouse provisionados

O código de exemplo abaixo mostra como:

- Criar um cluster do Redshift.
- Criar um perfil do IAM para acessar o S3.
- Criar tabelas e carregar dados.
- Executar consultas de exemplo.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon Redshift Provisioned Cluster Tutorial Script
# This script creates a Redshift cluster, loads sample data, runs queries, and
 cleans up resources
# Version 4: Security improvements and best practices

set -euo pipefail

# Set up logging
```

```
LOG_FILE="redshift_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Redshift tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to handle errors
handle_error() {
    echo "ERROR: $1" >&2
    echo "Resources created so far:"
    if [ -n "${CLUSTER_ID:-}" ]; then echo "- Redshift Cluster: $CLUSTER_ID"; fi
    if [ -n "${ROLE_NAME:-}" ]; then echo "- IAM Role: $ROLE_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete the cluster if it exists
    if [ -n "${CLUSTER_ID:-}" ]; then
        echo "Deleting Redshift cluster: $CLUSTER_ID"
        aws redshift delete-cluster --cluster-identifier "$CLUSTER_ID" --skip-
final-cluster-snapshot 2>/dev/null || true
        echo "Waiting for cluster deletion to complete..."
        aws redshift wait cluster-deleted --cluster-identifier "$CLUSTER_ID" 2>/
dev/null || true
        echo "Cluster deleted successfully."
    fi

    # Delete the IAM role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Removing IAM role policy..."
        aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 RedshiftS3Access 2>/dev/null || true

        echo "Deleting IAM role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    # Clean up temporary files
```

```
    rm -f redshift-trust-policy.json redshift-s3-policy.json

    echo "Cleanup completed."
}

# Trap errors and cleanup
trap 'handle_error "Script interrupted"' INT TERM

# Function to wait for SQL statement to complete
wait_for_statement() {
    local statement_id=$1
    local max_attempts=30
    local attempt=1
    local status=""

    echo "Waiting for statement $statement_id to complete..."

    while [ $attempt -le $max_attempts ]; do
        status=$(aws redshift-data describe-statement --id "$statement_id" --
query 'Status' --output text 2>/dev/null || echo "")

        if [ "$status" == "FINISHED" ]; then
            echo "Statement completed successfully."
            return 0
        elif [ "$status" == "FAILED" ]; then
            local error=$(aws redshift-data describe-statement --id
 "$statement_id" --query 'Error' --output text 2>/dev/null || echo "Unknown
 error")
            echo "Statement failed with error: $error" >&2
            return 1
        elif [ "$status" == "ABORTED" ]; then
            echo "Statement was aborted." >&2
            return 1
        fi

        echo "Statement status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 10
        ((attempt++))
    done

    echo "Timed out waiting for statement to complete." >&2
    return 1
}
```

```
# Function to check if IAM role is attached to cluster
check_role_attached() {
    local role_arn=$1
    local max_attempts=10
    local attempt=1

    echo "Checking if IAM role is attached to the cluster..."

    while [ $attempt -le $max_attempts ]; do
        local status=$(aws redshift describe-clusters \
            --cluster-identifier "$CLUSTER_ID" \
            --query "Clusters[0].IamRoles[?IamRoleArn=='$role_arn'].ApplyStatus"
 \
            --output text 2>/dev/null || echo "")

        if [ "$status" == "in-sync" ]; then
            echo "IAM role is successfully attached to the cluster."
            return 0
        fi

        echo "IAM role status: $status. Waiting... (Attempt $attempt/
$max_attempts)"
        sleep 30
        ((attempt++))
    done

    echo "Timed out waiting for IAM role to be attached." >&2
    return 1
}

# Validate required commands
for cmd in aws jq; do
    if ! command -v "$cmd" &> /dev/null; then
        handle_error "Required command '$cmd' not found. Please install it and
 try again."
    fi
done

# Validate AWS credentials
if ! aws sts get-caller-identity &>/dev/null; then
    handle_error "AWS credentials not configured or invalid"
fi
```

```
# Variables to track created resources
CLUSTER_ID="examplecluster"
ROLE_NAME="RedshiftS3Role-$(date +%s)"
DB_NAME="dev"
DB_USER="awsuser"

# Generate secure password using AWS Secrets Manager or random string
if command -v openssl &> /dev/null; then
    DB_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-20)
else
    DB_PASSWORD="TempPass$(date +%s | md5sum | cut -c1-20)"
fi

# Validate password meets requirements
if [ ${#DB_PASSWORD} -lt 8 ]; then
    handle_error "Generated password does not meet minimum length requirement"
fi

# Store password securely (optional: use AWS Secrets Manager in production)
echo "Generated database password (store securely): $DB_PASSWORD"

echo "=== Step 1: Creating Amazon Redshift Cluster ==="

# Create the Redshift cluster with encryption and audit logging enabled
echo "Creating Redshift cluster: $CLUSTER_ID"
CLUSTER_RESULT=$(aws redshift create-cluster \
  --cluster-identifier "$CLUSTER_ID" \
  --node-type ra3.xlplus \
  --number-of-nodes 2 \
  --master-username "$DB_USER" \
  --master-user-password "$DB_PASSWORD" \
  --db-name "$DB_NAME" \
  --port 5439 \
  --encrypted \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=redshift-provisioned \
  2>&1) || handle_error "Failed to create Redshift cluster"

echo "$CLUSTER_RESULT"
echo "Waiting for cluster to become available..."

# Wait for the cluster to be available
aws redshift wait cluster-available --cluster-identifier "$CLUSTER_ID" ||
 handle_error "Timeout waiting for cluster to become available"
```

```
# Get cluster status to confirm
CLUSTER_STATUS=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].ClusterStatus' \
  --output text)

echo "Cluster status: $CLUSTER_STATUS"

echo "=== Step 2: Creating IAM Role for S3 Access ==="

# Create trust policy file with restricted permissions
echo "Creating trust policy for Redshift"
cat > redshift-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

chmod 600 redshift-trust-policy.json

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://redshift-trust-policy.json 2>&1) ||
 handle_error "Failed to create IAM role"

echo "$ROLE_RESULT"

# Get the role ARN
ROLE_ARN=$(aws iam get-role --role-name "$ROLE_NAME" --query 'Role.Arn' --output
 text)
echo "Role ARN: $ROLE_ARN"

# Tag the IAM role
```

```
echo "Tagging IAM role: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=redshift-provisioned

# Create policy document for S3 access with principle of least privilege
echo "Creating S3 access policy"
cat > redshift-s3-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::redshift-downloads",
        "arn:aws:s3:::redshift-downloads/*"
      ]
    }
  ]
}
EOF

chmod 600 redshift-s3-policy.json

# Attach policy to role
echo "Attaching S3 access policy to role"
POLICY_RESULT=$(aws iam put-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-name RedshiftS3Access \
  --policy-document file://redshift-s3-policy.json 2>&1) || handle_error "Failed
 to attach policy to role"

echo "$POLICY_RESULT"

# Attach role to cluster
echo "Attaching IAM role to Redshift cluster"
ATTACH_ROLE_RESULT=$(aws redshift modify-cluster-iam-roles \
  --cluster-identifier "$CLUSTER_ID" \
  --add-iam-roles "$ROLE_ARN" 2>&1) || handle_error "Failed to attach role to
 cluster"
```

```
echo "$ATTACH_ROLE_RESULT"

# Wait for the role to be attached
echo "Waiting for IAM role to be attached to the cluster..."
if ! check_role_attached "$ROLE_ARN"; then
    handle_error "Failed to attach IAM role to cluster"
fi

echo "=== Step 3: Getting Cluster Connection Information ==="

# Get cluster endpoint
CLUSTER_INFO=$(aws redshift describe-clusters \
  --cluster-identifier "$CLUSTER_ID" \
  --query 'Clusters[0].Endpoint.{Address:Address,Port:Port}' \
  --output json)

echo "Cluster endpoint information:"
echo "$CLUSTER_INFO"

echo "=== Step 4: Creating Tables and Loading Data ==="

echo "Creating sales table"
SALES_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS sales; CREATE TABLE sales(salesid integer not null,
 listid integer not null distkey, sellerid integer not null, buyerid integer
 not null, eventid integer not null, dateid smallint not null sortkey, qtysold
 smallint not null, pricepaid decimal(8,2), commission decimal(8,2), saletime
 timestamp);" \
  --query 'Id' --output text)

echo "Sales table creation statement ID: $SALES_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_TABLE_ID"; then
    handle_error "Failed to create sales table"
fi

echo "Creating date table"
DATE_TABLE_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
```

```
  --db-user "$DB_USER" \
  --sql "DROP TABLE IF EXISTS date; CREATE TABLE date(dateid smallint not null
 distkey sortkey, caldate date not null, day character(3) not null, week smallint
 not null, month character(5) not null, qtr character(5) not null, year smallint
 not null, holiday boolean default('N'));" \
  --query 'Id' --output text)

echo "Date table creation statement ID: $DATE_TABLE_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_TABLE_ID"; then
    handle_error "Failed to create date table"
fi

echo "Loading data into sales table"
SALES_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY sales FROM 's3://redshift-downloads/tickit/sales_tab.txt' DELIMITER
 '\t' TIMEFORMAT 'MM/DD/YYYY HH:MI:SS' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';"
 \
  --query 'Id' --output text)

echo "Sales data load statement ID: $SALES_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$SALES_LOAD_ID"; then
    handle_error "Failed to load data into sales table"
fi

echo "Loading data into date table"
DATE_LOAD_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "COPY date FROM 's3://redshift-downloads/tickit/date2008_pipe.txt'
 DELIMITER '|' REGION 'us-east-1' IAM_ROLE '$ROLE_ARN';" \
  --query 'Id' --output text)

echo "Date data load statement ID: $DATE_LOAD_ID"

# Wait for statement to complete
if ! wait_for_statement "$DATE_LOAD_ID"; then
```

```
    handle_error "Failed to load data into date table"
fi

echo "=== Step 5: Running Example Queries ==="

echo "Running query: Get definition for the sales table"
QUERY1_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT * FROM pg_table_def WHERE tablename = 'sales';" \
  --query 'Id' --output text)

echo "Query 1 statement ID: $QUERY1_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY1_ID"; then
    handle_error "Query 1 failed"
fi

# Get and display results
echo "Query 1 results (first 10 rows):"
aws redshift-data get-statement-result --id "$QUERY1_ID" --max-items 10

echo "Running query: Find total sales on a given calendar date"
QUERY2_ID=$(aws redshift-data execute-statement \
  --cluster-identifier "$CLUSTER_ID" \
  --database "$DB_NAME" \
  --db-user "$DB_USER" \
  --sql "SELECT sum(qtysold) FROM sales, date WHERE sales.dateid = date.dateid
 AND caldate = '2008-01-05';" \
  --query 'Id' --output text)

echo "Query 2 statement ID: $QUERY2_ID"

# Wait for statement to complete
if ! wait_for_statement "$QUERY2_ID"; then
    handle_error "Query 2 failed"
fi

# Get and display results
echo "Query 2 results:"
aws redshift-data get-statement-result --id "$QUERY2_ID"
```

```
echo "=== Tutorial Complete ==="
echo "The following resources were created:"
echo "- Redshift Cluster: $CLUSTER_ID"
echo "- IAM Role: $ROLE_NAME"

echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Cleaning up all created resources..."
cleanup_resources
echo "All resources have been cleaned up."

# Securely clear password from memory
DB_PASSWORD=""

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - CreateCluster
  - CreateRole
  - DeleteCluster
  - DeleteRole
  - DeleteRolePolicy
  - DescribeClusters
  - GetRole
  - ModifyClusterIamRoles
  - PutRolePolicy
  - Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos da orquestração de fluxo de trabalaho

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM para o Step Functions.
- Criar sua primeira máquina de estado.
- Iniciar a execução da máquina de estado.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS Step Functions Getting Started Tutorial Script
# This script creates and runs a Step Functions state machine based on the AWS
 Step Functions Getting Started tutorial

set -euo pipefail

# Security: Restrict umask to prevent unintended file permissions
umask 077

# Parse command line arguments
AUTO_CLEANUP=true
while [[ $# -gt 0 ]]; do
    case $1 in
        --auto-cleanup)
            AUTO_CLEANUP=true
            shift
            ;;
        -h|--help)
            echo "Usage: $0 [--auto-cleanup] [--help]"
            echo "  --auto-cleanup: Automatically clean up resources without
 prompting"
            echo "  --help: Show this help message"
            exit 0
```

```
            ;;
        *)
            echo "Unknown option: $1"
            echo "Use --help for usage information"
            exit 1
            ;;
    esac
done

# Set up logging with secure permissions
LOG_FILE="step-functions-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"

# Security: Use process substitution with explicit FD cleanup
exec 3>&1 4>&2
exec > >(tee -a "$LOG_FILE") 2>&1
trap 'exec 1>&3 2>&4 3>&- 4>&-' EXIT

echo "Starting AWS Step Functions Getting Started Tutorial..."
echo "Logging to $LOG_FILE"

# Verify AWS CLI is installed and configured
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed"
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS credentials are not configured or invalid"
    exit 1
fi

# Check if jq is available for better JSON parsing
if ! command -v jq &> /dev/null; then
    echo "WARNING: jq is not installed. Using basic JSON parsing which may be
 less reliable."
    echo "Consider installing jq for better error handling: brew install jq
 (macOS) or apt-get install jq (Ubuntu)"
    USE_JQ=false
else
    USE_JQ=true
fi
```

```
# Use fixed region that supports Amazon Comprehend
CURRENT_REGION="us-west-2"
echo "Using fixed AWS region: $CURRENT_REGION (supports Amazon Comprehend)"

# Set AWS CLI to use the fixed region for all commands
export AWS_DEFAULT_REGION="$CURRENT_REGION"
export AWS_REGION="$CURRENT_REGION"

# Amazon Comprehend is available in us-west-2, so we can always enable it
echo "Amazon Comprehend is available in region $CURRENT_REGION"
SKIP_COMPREHEND=false

# Security: Initialize all resource variables
STATE_MACHINE_ARN=""
ROLE_NAME=""
ROLE_ARN=""
POLICY_ARN=""
STEPFUNCTIONS_POLICY_ARN=""
EXECUTION_ARN=""
EXECUTION2_ARN=""
EXECUTION3_ARN=""

# Performance: Cache for AWS API calls to reduce redundant requests
declare -A API_CACHE

# Function to make cached AWS CLI calls
aws_call_cached() {
    local cache_key="$1"
    shift

    if [[ -v API_CACHE["$cache_key"] ]]; then
        echo "${API_CACHE[$cache_key]}"
        return 0
    fi

    local result
    result=$(aws "$@" 2>&1) || return $?
    API_CACHE["$cache_key"]="$result"
    echo "$result"
}

# Function to check for API errors in JSON response with optimized jq usage
check_api_error() {
```

```
    local response="$1"
    local operation="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        # Use jq for more reliable JSON parsing with efficient error detection
        if echo "$response" | jq -e '.Error // .error // empty' > /dev/null 2>&1;
 then
            local error_message=$(echo "$response" | jq -r
 '.Error.Message // .Error.Code // .error // "Unknown error"' 2>/dev/null)
            handle_error "$operation failed: $error_message"
        fi
    else
        # Fallback to grep-based detection with optimized pattern
        if echo "$response" | grep -qE '"[Ee]rror":|"error":'; then
            handle_error "$operation failed: $response"
        fi
    fi
}

# Function to extract JSON field efficiently
extract_json_field() {
    local json="$1"
    local field="$2"

    if [[ "$USE_JQ" == "true" ]]; then
        echo "$json" | jq -r "$field" 2>/dev/null
    else
        echo "$json" | grep -oP "\"${field}\":\s*\"\K[^\"]+|\"${field}\":\s*
\K[^,}]+" | head -1
    fi
}

# Function to securely wait for resource propagation with exponential backoff
wait_for_propagation() {
    local resource_type="$1"
    local wait_time="${2:-10}"

    # Validate wait_time is a positive integer
    if ! [[ "$wait_time" =~ ^[0-9]+$ ]] || [ "$wait_time" -lt 1 ] ||
 [ "$wait_time" -gt 300 ]; then
        echo "WARNING: Invalid wait time $wait_time, using default 10 seconds"
        wait_time=10
    fi
```

```
    echo "Waiting for $resource_type to propagate ($wait_time seconds)..."
    sleep "$wait_time"
}

# Function to validate JSON file efficiently
validate_json_file() {
    local file="$1"

    if [[ "$USE_JQ" == "true" ]]; then
        if ! jq empty "$file" 2>/dev/null; then
            handle_error "Invalid JSON in $file"
        fi
    fi
}

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created:"
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "- State Machine: $STATE_MACHINE_ARN"
    fi
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "- IAM Role: $ROLE_NAME"
    fi
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "- IAM Policy: $POLICY_ARN"
    fi
    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "- Step Functions Policy: $STEPFUNCTIONS_POLICY_ARN"
    fi

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to securely clean up resources with parallel deletion
cleanup() {
    echo "Cleaning up resources..."

    # Delete state machine if it exists
    if [ -n "${STATE_MACHINE_ARN:-}" ]; then
        echo "Deleting state machine: $STATE_MACHINE_ARN"
```

```
        aws stepfunctions delete-state-machine --state-machine-arn
 "$STATE_MACHINE_ARN" 2>/dev/null &
    fi

    # Detach and delete policies if they exist
    if [ -n "${POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Comprehend policy $POLICY_ARN from role $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ] && [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching Step Functions policy $STEPFUNCTIONS_POLICY_ARN from role
 $ROLE_NAME"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/null &
    fi

    # Wait for detach operations to complete
    wait 2>/dev/null || true

    # Delete custom policies if they exist
    if [ -n "${POLICY_ARN:-}" ]; then
        echo "Deleting Comprehend policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" 2>/dev/null &
    fi

    if [ -n "${STEPFUNCTIONS_POLICY_ARN:-}" ]; then
        echo "Deleting Step Functions policy: $STEPFUNCTIONS_POLICY_ARN"
        aws iam delete-policy --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>/dev/
null &
    fi

    # Wait for policy deletion to complete
    wait 2>/dev/null || true

    # Delete role if it exists
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || echo "Failed
 to delete role"
    fi

    # Remove temporary files securely
```

```
    echo "Removing temporary files"
    local temp_files=(
        "hello-world.json"
        "updated-hello-world.json"
        "sentiment-hello-world.json"
        "step-functions-trust-policy.json"
        "comprehend-policy.json"
        "stepfunctions-policy.json"
        "input.json"
        "sentiment-input.json"
    )

    for file in "${temp_files[@]}"; do
        if [ -f "$file" ]; then
            if command -v shred &> /dev/null; then
                shred -vfz -n 3 "$file" 2>/dev/null || rm -f "$file"
            else
                rm -f "$file"
            fi
        fi
    done
}

# Security: Set trap to cleanup on script exit
trap cleanup EXIT

# Generate a secure random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
ROLE_NAME="StepFunctionsHelloWorldRole-${RANDOM_ID}"
POLICY_NAME="DetectSentimentPolicy-${RANDOM_ID}"
STATE_MACHINE_NAME="MyFirstStateMachine-${RANDOM_ID}"

echo "Using random identifier: $RANDOM_ID"
echo "Role name: $ROLE_NAME"
echo "Policy name: $POLICY_NAME"
echo "State machine name: $STATE_MACHINE_NAME"

# Step 1: Create the state machine definition
echo "Creating state machine definition..."
cat > hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
```

```
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Result": {
        "IsHelloWorldExample": true,
        "ExecutionWaitTimeInSeconds": 10
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
```

```
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Create IAM role trust policy
echo "Creating IAM role trust policy..."
cat > step-functions-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "states.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
```

```
}
EOF

# Create IAM role
echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://step-functions-trust-policy.json 2>&1)

check_api_error "$ROLE_RESULT" "Create IAM role"
echo "Role created successfully"
aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-step-functions-gs

# Get the role ARN
ROLE_ARN=$(extract_json_field "$ROLE_RESULT" ".Role.Arn")

if [ -z "$ROLE_ARN" ]; then
    handle_error "Failed to extract role ARN"
fi
echo "Role ARN: $ROLE_ARN"

# Create a custom policy for Step Functions with least privilege
echo "Creating custom policy for Step Functions..."
cat > stepfunctions-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "states:StartExecution",
        "states:DescribeExecution",
        "states:StopExecution"
      ],
      "Resource": "arn:aws:states:*:*:stateMachine:*"
    }
  ]
}
EOF

# Create the policy
echo "Creating Step Functions policy..."
STEPFUNCTIONS_POLICY_RESULT=$(aws iam create-policy \
```

```
  --policy-name "StepFunctionsPolicy-${RANDOM_ID}" \
  --policy-document file://stepfunctions-policy.json 2>&1)

check_api_error "$STEPFUNCTIONS_POLICY_RESULT" "Create Step Functions policy"
echo "Step Functions policy created successfully"

# Get the policy ARN
STEPFUNCTIONS_POLICY_ARN=$(extract_json_field "$STEPFUNCTIONS_POLICY_RESULT"
 ".Policy.Arn")

if [ -z "$STEPFUNCTIONS_POLICY_ARN" ]; then
    handle_error "Failed to extract Step Functions policy ARN"
fi
echo "Step Functions policy ARN: $STEPFUNCTIONS_POLICY_ARN"

# Attach policy to the role
echo "Attaching Step Functions policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$STEPFUNCTIONS_POLICY_ARN" 2>&1)

if [ $? -ne 0 ]; then
    handle_error "Failed to attach Step Functions policy to role: $ATTACH_RESULT"
fi

# Wait for role to propagate (IAM changes can take time to propagate)
wait_for_propagation "IAM role" 8

# Create state machine
echo "Creating state machine: $STATE_MACHINE_NAME"
SM_RESULT=$(aws stepfunctions create-state-machine \
  --name "$STATE_MACHINE_NAME" \
  --definition file://hello-world.json \
  --role-arn "$ROLE_ARN" \
  --type STANDARD \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-step-functions-gs
 2>&1)

check_api_error "$SM_RESULT" "Create state machine"
echo "State machine created successfully"

# Get the state machine ARN
STATE_MACHINE_ARN=$(extract_json_field "$SM_RESULT" ".stateMachineArn")
```

```
if [ -z "$STATE_MACHINE_ARN" ]; then
    handle_error "Failed to extract state machine ARN"
fi
echo "State machine ARN: $STATE_MACHINE_ARN"

# Step 2: Start the state machine execution
echo "Starting state machine execution..."
EXEC_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello001-${RANDOM_ID}" 2>&1)

check_api_error "$EXEC_RESULT" "Start execution"
echo "Execution started successfully"

# Get the execution ARN
EXECUTION_ARN=$(extract_json_field "$EXEC_RESULT" ".executionArn")

if [ -z "$EXECUTION_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
echo "Execution ARN: $EXECUTION_ARN"

# Wait for execution to complete (the workflow has a 10-second wait state)
echo "Waiting for execution to complete (12 seconds)..."
sleep 12

# Check execution status
echo "Checking execution status..."
EXEC_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION_ARN" 2>&1)

echo "Execution status: $EXEC_STATUS"

# Step 3: Update state machine to process external input
echo "Updating state machine to process external input..."
cat > updated-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example of the Amazon States Language using a Pass
 state",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
```

```
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "FailState"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
              },
```

```
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailState": {
      "Type": "Fail",
      "Error": "NotHelloWorldExample",
      "Cause": "The IsHelloWorldExample value was false"
    }
  }
}
EOF

# Update state machine
echo "Updating state machine..."
UPDATE_RESULT=$(aws stepfunctions update-state-machine \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --definition file://updated-hello-world.json \
  --role-arn "$ROLE_ARN" 2>&1)

check_api_error "$UPDATE_RESULT" "Update state machine"
echo "State machine updated successfully"

# Create input file with strict validation
echo "Creating input file..."
cat > input.json << 'EOF'
{
  "wait": 5,
  "hello_world": true
}
EOF
```

```
# Validate input JSON
validate_json_file "input.json"

# Start execution with input
echo "Starting execution with input..."
EXEC2_RESULT=$(aws stepfunctions start-execution \
  --state-machine-arn "$STATE_MACHINE_ARN" \
  --name "hello002-${RANDOM_ID}" \
  --input file://input.json 2>&1)

check_api_error "$EXEC2_RESULT" "Start execution with input"
echo "Execution with input started successfully"

# Get the execution ARN
EXECUTION2_ARN=$(extract_json_field "$EXEC2_RESULT" ".executionArn")

if [ -z "$EXECUTION2_ARN" ]; then
    handle_error "Failed to extract execution ARN"
fi
echo "Execution ARN: $EXECUTION2_ARN"

# Wait for execution to complete (the workflow has a 5-second wait state)
echo "Waiting for execution to complete (8 seconds)..."
sleep 8

# Check execution status
echo "Checking execution status..."
EXEC2_STATUS=$(aws stepfunctions describe-execution \
  --execution-arn "$EXECUTION2_ARN" 2>&1)

echo "Execution status: $EXEC2_STATUS"

# Step 4: Integrate Amazon Comprehend for sentiment analysis (if available)
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Creating policy for Amazon Comprehend access with least privilege..."
    cat > comprehend-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "comprehend:DetectSentiment"
```

```
      ],
      "Resource": "*"
    }
  ]
}
EOF

    # Create policy
    echo "Creating IAM policy: $POLICY_NAME"
    POLICY_RESULT=$(aws iam create-policy \
      --policy-name "$POLICY_NAME" \
      --policy-document file://comprehend-policy.json 2>&1)

    check_api_error "$POLICY_RESULT" "Create Comprehend policy"
    echo "Comprehend policy created successfully"

    # Get policy ARN
    POLICY_ARN=$(extract_json_field "$POLICY_RESULT" ".Policy.Arn")

    if [ -z "$POLICY_ARN" ]; then
        handle_error "Failed to extract policy ARN"
    fi
    echo "Policy ARN: $POLICY_ARN"

    # Attach policy to role
    echo "Attaching policy to role..."
    ATTACH2_RESULT=$(aws iam attach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN" 2>&1)

    if [ $? -ne 0 ]; then
        handle_error "Failed to attach policy to role: $ATTACH2_RESULT"
    fi

    # Create updated state machine definition with sentiment analysis
    echo "Creating updated state machine definition with sentiment analysis..."
    cat > sentiment-hello-world.json << 'EOF'
{
  "Comment": "A Hello World example with sentiment analysis",
  "StartAt": "SetVariables",
  "States": {
    "SetVariables": {
      "Type": "Pass",
      "Parameters": {
```

```
        "IsHelloWorldExample.$": "$.hello_world",
        "ExecutionWaitTimeInSeconds.$": "$.wait",
        "FeedbackComment.$": "$.feedback_comment"
      },
      "Next": "IsHelloWorldExample"
    },
    "IsHelloWorldExample": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "$.IsHelloWorldExample",
          "BooleanEquals": true,
          "Next": "WaitState"
        }
      ],
      "Default": "DetectSentiment"
    },
    "WaitState": {
      "Type": "Wait",
      "SecondsPath": "$.ExecutionWaitTimeInSeconds",
      "Next": "ParallelProcessing"
    },
    "ParallelProcessing": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "Process1",
          "States": {
            "Process1": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 1"
              },
              "End": true
            }
          }
        },
        {
          "StartAt": "Process2",
          "States": {
            "Process2": {
              "Type": "Pass",
              "Result": {
                "message": "Processing task 2"
```

```
              },
              "End": true
            }
          }
        }
      ],
      "Next": "CheckpointState"
    },
    "CheckpointState": {
      "Type": "Pass",
      "Result": {
        "CheckpointMessage": "Workflow completed successfully!"
      },
      "Next": "SuccessState"
    },
    "DetectSentiment": {
      "Type": "Task",
      "Resource": "arn:aws:states:::aws-sdk:comprehend:detectSentiment",
      "Parameters": {
        "LanguageCode": "en",
        "Text.$": "$.FeedbackComment"
      },
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    }
  }
}
EOF

    # Validate sentiment state machine JSON
    validate_json_file "sentiment-hello-world.json"

    # Wait for IAM changes to propagate
    wait_for_propagation "IAM changes" 8

    # Update state machine
    echo "Updating state machine with sentiment analysis..."
    UPDATE2_RESULT=$(aws stepfunctions update-state-machine \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --definition file://sentiment-hello-world.json \
      --role-arn "$ROLE_ARN" 2>&1)
```

```
    check_api_error "$UPDATE2_RESULT" "Update state machine with sentiment
 analysis"
    echo "State machine updated with sentiment analysis successfully"

    # Create input file with feedback comment
    echo "Creating input file with feedback comment..."
    cat > sentiment-input.json << 'EOF'
{
  "hello_world": false,
  "wait": 5,
  "feedback_comment": "This getting started with Step Functions workshop is a
 challenge!"
}
EOF

    # Validate sentiment input JSON
    validate_json_file "sentiment-input.json"

    # Start execution with sentiment analysis input
    echo "Starting execution with sentiment analysis input..."
    EXEC3_RESULT=$(aws stepfunctions start-execution \
      --state-machine-arn "$STATE_MACHINE_ARN" \
      --name "hello003-${RANDOM_ID}" \
      --input file://sentiment-input.json 2>&1)

    check_api_error "$EXEC3_RESULT" "Start execution with sentiment analysis"
    echo "Execution with sentiment analysis started successfully"

    # Get the execution ARN
    EXECUTION3_ARN=$(extract_json_field "$EXEC3_RESULT" ".executionArn")

    if [ -z "$EXECUTION3_ARN" ]; then
        handle_error "Failed to extract execution ARN"
    fi
    echo "Execution ARN: $EXECUTION3_ARN"

    # Wait for execution to complete
    echo "Waiting for execution to complete (3 seconds)..."
    sleep 3

    # Check execution status
    echo "Checking execution status..."
    EXEC3_STATUS=$(aws stepfunctions describe-execution \
      --execution-arn "$EXECUTION3_ARN" 2>&1)
```

```
    echo "Execution status: $EXEC3_STATUS"
else
    echo "Skipping Amazon Comprehend integration (not available in
 $CURRENT_REGION)"
    EXECUTION3_ARN=""
fi

# Display summary of resources created
echo ""
echo "============================================="
echo "RESOURCES CREATED"
echo "============================================="
echo "State Machine: $STATE_MACHINE_ARN"
echo "IAM Role: $ROLE_NAME"
echo "Step Functions Policy: StepFunctionsPolicy-${RANDOM_ID}
 ($STEPFUNCTIONS_POLICY_ARN)"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "Comprehend Policy: $POLICY_NAME ($POLICY_ARN)"
fi
echo "Executions:"
echo "  - hello001-${RANDOM_ID}: $EXECUTION_ARN"
echo "  - hello002-${RANDOM_ID}: $EXECUTION2_ARN"
if [[ "$SKIP_COMPREHEND" == "false" ]]; then
    echo "  - hello003-${RANDOM_ID}: $EXECUTION3_ARN"
fi
echo "============================================="

# Cleanup
echo ""
echo "============================================="
echo "CLEANUP"
echo "============================================="
echo "Auto-cleanup enabled. Cleaning up resources..."

echo "All resources have been cleaned up."

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreatePolicy

- CreateRole
- CreateStateMachine
- DeletePolicy
- DeleteRole
- DeleteStateMachine
- DescribeExecution
- DetachRolePolicy
- StartExecution
- UpdateStateMachine

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Gerenciar chaves de acesso do IAM usando um AWS SDK

O exemplo de código a seguir mostra como gerenciar chaves de acesso.

> **Warning**
>
> Para evitar riscos de segurança, não use usuários do IAM para autenticação ao desenvolver software com propósito específico ou trabalhar com dados reais. Em vez disso, use federação com um provedor de identidade, como Centro de Identidade do AWS IAM.

- Criar e listar chaves de acesso.
- Descobrir quando e como a chave de acesso foi usada pela última vez.
- Atualizar e excluir chaves de acesso.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações de chave de acesso do IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.
```

```
    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Use a funções de wrapper para executar ações de chave de acesso para o usuário atual.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
 {new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
 {last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
```

```
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - CreateAccessKey
  - DeleteAccessKey
  - GetAccessKeyLastUsed
  - ListAccessKeys
  - UpdateAccessKey

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Gerenciar políticas do IAM usando um AWS SDK

O código de exemplo abaixo mostra como:

- Criar e listar políticas.
- Criar e obter versões de políticas.
- Reverter uma política para uma versão anterior.
- Excluir políticas.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie a funções que envolvam ações de política do IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::amzn-s3-demo-bucket/*' to allow actions on
 all objects
                         in the bucket named 'amzn-s3-demo-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version":"2012-10-17",
```

```
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
```

```
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version":"2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
```

```
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
 rollback_version.version_id)
        default_version.delete()
```

```
        logger.info("Deleted original default version %s.",
 default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
 so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Use a funções de wrapper para criar políticas, atualizar versões e obter informações sobre elas.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
```

```
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::amzn-s3-demo-bucket"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Gerenciar perfis do IAM usando um AWS SDK

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM.
- Anexar e separar políticas para um perfil.
- Excluir um perfil.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Criar perfis que envolvam ações de perfil do IAM.

```
import json
```

```
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
```

```
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
```

```
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Use a funções de wrapper para criar uma função, depois, anexe e desanexe uma política.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
 "batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - AttachRolePolicy
  - CreateRole
  - DeleteRole
  - DetachRolePolicy

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Gerenciar a conta do IAM usando um AWS SDK

O código de exemplo abaixo mostra como:

- Obter e atualizar o alias da conta.
- Gerar um relatório de usuários e credenciais.
- Obter um resumo da utilização da conta.
- Obtenha detalhes de todos os usuários, grupos, perfis e políticas em sua conta, incluindo as relações uns com os outros.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie funções que envolvam ações de conta do IAM.

```
import logging
import pprint
import sys
import time
```

```
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]



def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
```

```
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
```

```
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Chame funções de wrapper para alterar o alias da conta e obter relatórios sobre a conta.

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
```

```
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
```

```
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialReport
  - GetAccountAuthorizationDetails
  - GetAccountSummary
  - GetCredentialReport
  - ListAccountAliases

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como mover segredos codificados para o armazenamento seguro de segredos

O código de exemplo abaixo mostra como:

- Criar perfis do IAM.
- Criar um segredo no Secrets Manager.
- Atualizar o código de uma aplicação.
- Atualizar o segredo.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail

# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"
```

```
# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "========================================="
    echo "RESOURCES CREATED"
    echo "========================================="

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi

    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
```

```
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
```

```
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
```

```
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"
```

```
# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)
```

```
check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"
```

```
# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "======================================="
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi

echo ""
echo "======================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""
```

```
echo "Script completed at $(date)"
exit 0
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity
  - GetSecretValue
  - PutResourcePolicy
  - UpdateSecret

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Permite que o atributo de automação AWS Compute Optimizer aplique as ações recomendadas

O exemplo de código a seguir mostra como. Esta política baseada em permissões permite que o atributo de automação AWS Compute Optimizer aplique as ações recomendadas

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "aco-automation.amazonaws.com"
            },
```

```
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para permitir a automação em toda a sua organização

O exemplo de código a seguir mostra como esta política baseada em permissões permite a automação em toda a sua organização

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": "aco-automation:AssociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:DisassociateAccounts",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:ListAccounts",
            "Resource": "*"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para permitir a automação na sua conta

O exemplo de código a seguir mostra como esta política baseada em permissões permite a automação na sua conta

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        },
        {
            "Effect": "Allow",
            "Action": "aco-automation:UpdateEnrollmentConfiguration",
            "Resource": "*"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para conceder acesso total ao Compute Optimizer Automation para uma conta gerencial de uma organização

O exemplo de código a seguir mostra como esta política baseada em permissões concede acesso total à automação do Compute Optimizer Automation para uma conta gerencial de uma organização

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:*",
               "ec2:DescribeVolumes",
               "organizations:ListAccounts",
               "organizations:DescribeOrganization",
               "organizations:DescribeAccount",
               "organizations:EnableAWSServiceAccess",
```

```
                "organizations:ListDelegatedAdministrators",
                "organizations:RegisterDelegatedAdministrator",
                "organizations:DeregisterDelegatedAdministrator"
            ],
            "Resource": "*"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para conceder acesso total ao Compute Optimizer Automation para contas autônomas da AWS

O exemplo de código a seguir mostra como esta política baseada em permissões concede acesso total ao Compute Optimizer Automation para contas autônomas da AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aco-automation:*",
            "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para conceder acesso somente leitura ao Compute Optimizer Automation para uma conta gerencial de uma organização

O exemplo de código a seguir mostra como esta política baseada em permissões concede acesso somente leitura ao Compute Optimizer Automation para uma conta gerencial de uma organização

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:GetEnrollmentConfiguration",
               "aco-automation:GetAutomationEvent",
               "aco-automation:GetAutomationRule",
               "aco-automation:ListAccounts",
               "aco-automation:ListAutomationEvents",
               "aco-automation:ListAutomationEventSteps",
               "aco-automation:ListAutomationEventSummaries",
               "aco-automation:ListAutomationRules",
               "aco-automation:ListAutomationRulePreview",
               "aco-automation:ListAutomationRulePreviewSummaries",
               "aco-automation:ListRecommendedActions",
               "aco-automation:ListRecommendedActionSummaries",
               "aco-automation:ListTagsForResource",
               "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para conceder acesso somente leitura ao Compute Optimizer Automation para contas autônomas da AWS

O exemplo de código a seguir mostra como esta política baseada em permissões concede acesso somente leitura ao Compute Optimizer Automation para contas autônomas da AWS

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "aco-automation:GetEnrollmentConfiguration",
               "aco-automation:GetAutomationEvent",
               "aco-automation:GetAutomationRule",
               "aco-automation:ListAutomationEvents",
               "aco-automation:ListAutomationEventSteps",
               "aco-automation:ListAutomationEventSummaries",
               "aco-automation:ListAutomationRules",
               "aco-automation:ListAutomationRulePreview",
               "aco-automation:ListAutomationRulePreviewSummaries",
               "aco-automation:ListRecommendedActions",
               "aco-automation:ListRecommendedActionSummaries",
               "aco-automation:ListTagsForResource",
               "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Política para conceder permissões de perfil vinculado ao serviço para o Compute Optimization Automation

O exemplo de código a seguir mostra como esta política baseada em permissões concede permissões de perfil vinculado ao serviço para o Compute Optimization Automation

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation",
            "Condition": {"StringLike": {"iam:AWSServiceName": "aco-
automation.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": "iam:PutRolePolicy",
            "Resource": "arn:aws:iam::*:role/aws-service-role/aco-
automation.amazonaws.com/AWSServiceRoleForComputeOptimizerAutomation"
        }
    ]
}
```

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Reverter uma versão de política do IAM usando um AWS SDK

O código de exemplo abaixo mostra como:

- Obter a lista de versões da política em ordem por data.
- Encontrar a versão da política padrão.
- Tornar a versão da política anterior a padrão.

- Excluir a versão padrão antiga.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
```

```
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

O código de exemplo abaixo mostra como:

- Criar perfis do IAM.
- Criar um alarme do CloudWatch.
- Criar um modelo de experimento
- Executar o experimento.
- Verificar os resultados.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
```

```
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true
```

```
        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi

    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
```

```
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
```

```
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
```

```
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"
```

```
# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60

# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
```

```
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
```

```
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
```

```
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail

    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
```

```
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
```

```
    echo "- $resource"
done
echo "========================================="

# Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
 "$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
 "$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
```

```
    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi
```

```
echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy
  - GetCallerIdentity
  - GetExperiment
  - GetMetricData
  - GetRole
  - MonitorInstances
  - PutMetricAlarm
  - PutRolePolicy
  - RemoveRoleFromInstanceProfile
  - RunInstances
  - StartExperiment
  - StopExperiment
  - TerminateInstances

- Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Configurar o controle de acesso por atributo para o DynamoDB usando a AWS Command Line Interface v2

O exemplo de código apresentado a seguir demonstra como implementar o controle de acesso por atributo (ABAC) no DynamoDB.

- Crie uma política do IAM para o ABAC.
- Crie tabelas com etiquetas que representem diferentes departamentos.
- Liste e filtre tabelas com base nas etiquetas.

Bash

AWS CLI com script Bash

Crie uma política do IAM para o ABAC.

```
# Step 1: Create a policy document for ABAC
cat > abac-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:Scan"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}"
        }
```

```
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": "arn:aws:dynamodb:*:*:table/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Department": "${aws:PrincipalTag/Department}",
          "aws:ResourceTag/Environment": "Development"
        }
      }
    }
  ]
}
EOF

# Step 2: Create the IAM policy
aws iam create-policy \
    --policy-name DynamoDBDepartmentBasedAccess \
    --policy-document file://abac-policy.json
```

Crie tabelas com etiquetas que representem diferentes departamentos.

```
# Create a DynamoDB table with tags for ABAC
aws dynamodb create-table \
    --table-name FinanceData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Finance \
        Key=Environment,Value=Development

# Create another table with different tags
```

```
aws dynamodb create-table \
    --table-name MarketingData \
    --attribute-definitions \
        AttributeName=RecordID,AttributeType=S \
    --key-schema \
        AttributeName=RecordID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --tags \
        Key=Department,Value=Marketing \
        Key=Environment,Value=Production
```

Liste e filtre tabelas com base nas etiquetas.

```
# List all DynamoDB tables
echo "Listing all tables:"
aws dynamodb list-tables

# Get ARNs for all tables
echo -e "\nGetting ARNs for all tables:"
TABLE_ARNS=$(aws dynamodb list-tables --query "TableNames[*]" --output text |
 xargs -I {} aws dynamodb describe-table --table-name {} --query "Table.TableArn"
 --output text)

# For each table ARN, list its tags
echo -e "\nListing tags for each table:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    echo -e "\nTags for table: $TABLE_NAME"
    aws dynamodb list-tags-of-resource --resource-arn $ARN
done

# Example: Find tables with a specific tag
echo -e "\nFinding tables with Environment=Production tag:"
for ARN in $TABLE_ARNS; do
    TABLE_NAME=$(echo $ARN | awk -F/ '{print $2}')
    TAGS=$(aws dynamodb list-tags-of-resource --resource-arn $ARN --query "Tags[?
Key=='Environment' && Value=='Production']" --output text)
    if [ ! -z "$TAGS" ]; then
        echo "Table with Production tag: $TABLE_NAME"
    fi
done
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - CreatePolicy
  - CreateTable
  - ListTables

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como configurar o gerenciamento de sistemas

O código de exemplo abaixo mostra como:

- Criar permissões do IAM para o Systems Manager.
- Criar um perfil do IAM para o Systems Manager
- Configurar o Systems Manager.
- Verificar a configuração.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
```

```
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}

# Function to clean up resources on error
```

```
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
            "SSM_CONFIG_MANAGER")
```

```
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done

    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
```

```
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
         "ssm:ListResourceDataSync",
         "ssm:DescribePatchBaselines",
         "ssm:GetPatchBaseline",
         "ssm:DescribeMaintenanceWindows",
         "ssm:DescribeMaintenanceWindowTasks"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmDocument",
       "Effect": "Allow",
       "Action": [
         "ssm:GetDocument",
         "ssm:DescribeDocument"
       ],
       "Resource": [
         "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
         "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
       ]
     },
     {
       "Sid": "SsmEnableExplorer",
       "Effect": "Allow",
       "Action": "ssm:StartAutomationExecution",
```

```
      "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
      "Sid": "SsmExplorerRds",
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsSummary",
        "ssm:CreateResourceDataSync",
        "ssm:UpdateResourceDataSync"
      ],
      "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
    },
    {
      "Sid": "OrgsReadOnly",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListRoots",
        "organizations:ListParents",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgsAdministration",
      "Effect": "Allow",
      "Action": [
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "organizations:ServicePrincipal": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "member.org.stacksets.cloudformation.amazonaws.com",
            "resource-explorer-2.amazonaws.com"
          ]
```

```
        }
      }
    },
    {
      "Sid": "CfnReadOnly",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ListStacks",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackSets",
        "cloudformation:DescribeOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgCfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ActivateOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CfnStackActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:RollbackStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStack"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
      ]
    },
    {
      "Sid": "CfnStackSetActions",
```

```
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:CreateStackSet",
        "cloudformation:DeleteStackInstances",
        "cloudformation:DeleteStackSet",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
```

```
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ]
      },
      {
        "Sid": "IamPassRole",
        "Effect": "Allow",
        "Action": [
          "iam:PassRole"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ],
        "Condition": {
          "StringEquals": {
            "iam:PassedToService": [
              "ssm.amazonaws.com",
              "ssm-quicksetup.amazonaws.com",
              "cloudformation.amazonaws.com"
            ]
          }
        }
      },
      {
        "Sid": "IamRolesPoliciesMgmt",
        "Effect": "Allow",
        "Action": [
          "iam:AttachRolePolicy",
          "iam:DetachRolePolicy"
        ],
        "Resource": [
          "arn:aws:iam::*:role/AWS-QuickSetup-*",
          "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
        ],
        "Condition": {
          "ArnEquals": {
            "iam:PolicyARN": [
              "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
              "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
            ]
```

```
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
```

```
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
```

```
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json

# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"

echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"
```

```
# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created configuration manager
```

```
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "==========================================="
echo "CREATED RESOURCES"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
```

```
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy
  - CreateRole
  - DeleteConfigurationManager
  - DeletePolicy
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetConfigurationManager

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como usar variáveis de propriedade nos painéis de monitoramento para monitorar várias funções com tecnologia sem servidor

O código de exemplo abaixo mostra como:

- Criar funções do Lambda para monitoramento.
- Criar um painel do CloudWatch
- Adicionar uma variável de propriedade ao painel.

- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# CloudWatch Dashboard with Lambda Function Variable Script
# This script creates a CloudWatch dashboard with a property variable for Lambda
 function names

set -euo pipefail

# Security: Set restrictive umask
umask 0077

# Set up logging with secure permissions
LOG_FILE="cloudwatch-dashboard-script-v4.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
echo "Starting script execution at $(date)" >> "$LOG_FILE"

# Function to log commands and their output (with sensitive data sanitization)
log_cmd() {
    local cmd="$1"
    local sanitized_cmd="${cmd//--password*/--password [REDACTED]}"
    sanitized_cmd="${sanitized_cmd//--secret*/--secret [REDACTED]}"
    echo "$(date): Running command: $sanitized_cmd" >> "$LOG_FILE"
    eval "$cmd" 2>&1 | tee -a "$LOG_FILE"
    return ${PIPESTATUS[0]}
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
```

```
    local cmd_status="$2"
    local error_msg="$3"

    if [ $cmd_status -ne 0 ] || echo "$cmd_output" | grep -qi "error"; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        # Sanitize output before logging
        local sanitized_output="${cmd_output//arn:aws:iam::[0-9]*/
arn:aws:iam::ACCOUNT_ID}"
        echo "Command output: $sanitized_output" | tee -a "$LOG_FILE"
        cleanup_resources
        exit 1
    fi
}

# Trap errors and cleanup
trap 'cleanup_resources' EXIT ERR INT TERM

# Function to clean up resources
cleanup_resources() {
    local exit_code=$?

    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "CLEANUP PROCESS" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"

    if [ -n "${DASHBOARD_NAME:-}" ]; then
        echo "Deleting CloudWatch dashboard: $DASHBOARD_NAME" | tee -a
 "$LOG_FILE"
        aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" 2>&1
 >> "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION1:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION1" 2>&1 >>
 "$LOG_FILE" || true
    fi

    if [ -n "${LAMBDA_FUNCTION2:-}" ]; then
        echo "Deleting Lambda function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
        aws lambda delete-function --function-name "$LAMBDA_FUNCTION2" 2>&1 >>
 "$LOG_FILE" || true
    fi
```

```
    if [ -n "${ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole 2>&1 >>
 "$LOG_FILE" || true

        echo "Deleting IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
        aws iam delete-role --role-name "$ROLE_NAME" 2>&1 >> "$LOG_FILE" || true
    fi

    # Clean up temporary files securely
    shred -vfz -n 3 trust-policy.json lambda_function.py lambda_function.zip 2>/
dev/null || rm -f trust-policy.json lambda_function.py lambda_function.zip

    echo "Cleanup completed." | tee -a "$LOG_FILE"

    return $exit_code
}

# Validate AWS CLI is installed and authenticated
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed" | tee -a "$LOG_FILE"
    exit 1
fi

if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not properly authenticated" | tee -a "$LOG_FILE"
    exit 1
fi

# Get AWS region with validation
AWS_REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION" | tee -a
 "$LOG_FILE"
else
    echo "Using AWS region: $AWS_REGION" | tee -a "$LOG_FILE"
fi

# Validate region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    echo "ERROR: Invalid AWS region format: $AWS_REGION" | tee -a "$LOG_FILE"
```

```
    exit 1
fi

# Generate unique identifiers using secure random with validation
RANDOM_ID=$(openssl rand -hex 6)
if [ -z "$RANDOM_ID" ] || [ ${#RANDOM_ID} -ne 12 ]; then
    echo "ERROR: Failed to generate valid random identifier" | tee -a "$LOG_FILE"
    exit 1
fi

DASHBOARD_NAME="LambdaMetricsDashboard-${RANDOM_ID}"
LAMBDA_FUNCTION1="TestFunction1-${RANDOM_ID}"
LAMBDA_FUNCTION2="TestFunction2-${RANDOM_ID}"
ROLE_NAME="LambdaExecutionRole-${RANDOM_ID}"

# Validate resource names don't exceed AWS limits
if [ ${#DASHBOARD_NAME} -gt 128 ] || [ ${#LAMBDA_FUNCTION1} -gt 64 ] ||
 [ ${#ROLE_NAME} -gt 64 ]; then
    echo "ERROR: Generated resource names exceed AWS limits" | tee -a "$LOG_FILE"
    exit 1
fi

echo "Using random identifier: $RANDOM_ID" | tee -a "$LOG_FILE"
echo "Dashboard name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda function names: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "IAM role name: $ROLE_NAME" | tee -a "$LOG_FILE"

# Create IAM role for Lambda functions
echo "Creating IAM role for Lambda..." | tee -a "$LOG_FILE"
TRUST_POLICY='{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}'

echo "$TRUST_POLICY" > trust-policy.json
```

```
chmod 600 trust-policy.json

# Validate JSON before use
if ! python3 -m json.tool trust-policy.json > /dev/null 2>&1; then
    echo "ERROR: Invalid trust policy JSON" | tee -a "$LOG_FILE"
    exit 1
fi

ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name '$ROLE_NAME' --assume-
role-policy-document file://trust-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=cloudwatch-streams --output json")
check_error "$ROLE_OUTPUT" $? "Failed to create IAM role"

ROLE_ARN=$(echo "$ROLE_OUTPUT" | python3 -c "import sys, json;
 print(json.load(sys.stdin)['Role']['Arn'])" 2>/dev/null)
if [ -z "$ROLE_ARN" ]; then
    echo "ERROR: Failed to extract Role ARN" | tee -a "$LOG_FILE"
    exit 1
fi
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Attach Lambda basic execution policy to the role
echo "Attaching Lambda execution policy to role..." | tee -a "$LOG_FILE"
POLICY_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name '$ROLE_NAME' --
policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole")
check_error "$POLICY_OUTPUT" $? "Failed to attach policy to role"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..." | tee -a "$LOG_FILE"
sleep 10

# Create simple Python Lambda function code with security validation
echo "Creating Lambda function code..." | tee -a "$LOG_FILE"
cat > lambda_function.py << 'LAMBDA_EOF'
def handler(event, context):
    print("Lambda function executed successfully")
    return {
        'statusCode': 200,
        'body': 'Success'
    }
LAMBDA_EOF

chmod 600 lambda_function.py
```

```
# Validate Python syntax
if ! python3 -m py_compile lambda_function.py 2>/dev/null; then
    echo "ERROR: Invalid Python syntax in Lambda function" | tee -a "$LOG_FILE"
    exit 1
fi

# Zip the Lambda function code
zip -j -q lambda_function.zip lambda_function.py
if [ ! -f lambda_function.zip ]; then
    echo "ERROR: Failed to create lambda_function.zip" | tee -a "$LOG_FILE"
    exit 1
fi
chmod 600 lambda_function.zip

# Validate zip file integrity
if ! unzip -t lambda_function.zip > /dev/null 2>&1; then
    echo "ERROR: Created zip file is corrupted" | tee -a "$LOG_FILE"
    exit 1
fi

# Create first Lambda function
echo "Creating first Lambda function: $LAMBDA_FUNCTION1..." | tee -a "$LOG_FILE"
LAMBDA1_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION1' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA1_OUTPUT" $? "Failed to create first Lambda function"

# Create second Lambda function
echo "Creating second Lambda function: $LAMBDA_FUNCTION2..." | tee -a "$LOG_FILE"
LAMBDA2_OUTPUT=$(log_cmd "aws lambda create-function --function-name
 '$LAMBDA_FUNCTION2' --runtime python3.11 --role '$ROLE_ARN' --handler
 lambda_function.handler --zip-file fileb://lambda_function.zip --timeout 30 --
memory-size 128 --tags project=doc-smith,tutorial=cloudwatch-streams")
check_error "$LAMBDA2_OUTPUT" $? "Failed to create second Lambda function"

# Invoke Lambda functions to generate some metrics
echo "Invoking Lambda functions to generate metrics..." | tee -a "$LOG_FILE"
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION1' --payload '{}' /
dev/null" || true
log_cmd "aws lambda invoke --function-name '$LAMBDA_FUNCTION2' --payload '{}' /
dev/null" || true

# Create CloudWatch dashboard with property variable
```

```
echo "Creating CloudWatch dashboard with property variable..." | tee -a
 "$LOG_FILE"

# Create dashboard body with proper escaping and validation
DASHBOARD_BODY=$(cat <<EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "$LAMBDA_FUNCTION1" ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$AWS_REGION",
        "title": "Lambda Invocations",
        "period": 300,
        "stat": "Sum"
      }
    }
  ]
}
EOF
)

# Validate JSON before sending
if ! echo "$DASHBOARD_BODY" | python3 -m json.tool > /dev/null 2>&1; then
    echo "ERROR: Dashboard body is not valid JSON" | tee -a "$LOG_FILE"
    exit 1
fi

# First create a basic dashboard without variables
echo "Creating initial dashboard without variables..." | tee -a "$LOG_FILE"
DASHBOARD_OUTPUT=$(aws cloudwatch put-dashboard --dashboard-name
 "$DASHBOARD_NAME" --dashboard-body "$DASHBOARD_BODY" --output json 2>&1)
check_error "$DASHBOARD_OUTPUT" $? "Failed to create initial CloudWatch
 dashboard"

# Now let's try to add a property variable using the console instructions
```

```
echo "To complete the tutorial, please follow these steps in the CloudWatch
 console:" | tee -a "$LOG_FILE"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/" | tee -a "$LOG_FILE"
echo "2. Navigate to Dashboards and select your dashboard: $DASHBOARD_NAME" | tee
 -a "$LOG_FILE"
echo "3. Choose Actions > Variables > Create a variable" | tee -a "$LOG_FILE"
echo "4. Choose Property variable" | tee -a "$LOG_FILE"
echo "5. For Property that the variable changes, choose FunctionName" | tee -a
 "$LOG_FILE"
echo "6. For Input type, choose Select menu (dropdown)" | tee -a "$LOG_FILE"
echo "7. Choose Use the results of a metric search" | tee -a "$LOG_FILE"
echo "8. Choose Pre-built queries > Lambda > Errors" | tee -a "$LOG_FILE"
echo "9. Choose By Function Name and then choose Search" | tee -a "$LOG_FILE"
echo "10. (Optional) Configure any secondary settings as desired" | tee -a
 "$LOG_FILE"
echo "11. Choose Add variable" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "The dashboard has been created and can be accessed at:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"

# Verify dashboard creation
echo "Verifying dashboard creation..." | tee -a "$LOG_FILE"
VERIFY_OUTPUT=$(aws cloudwatch get-dashboard --dashboard-name "$DASHBOARD_NAME"
 --output json 2>&1)
check_error "$VERIFY_OUTPUT" $? "Failed to verify dashboard creation"

echo "" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "DASHBOARD CREATED SUCCESSFULLY" | tee -a "$LOG_FILE"
echo "==========================================" | tee -a "$LOG_FILE"
echo "Dashboard Name: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "Lambda Functions: $LAMBDA_FUNCTION1, $LAMBDA_FUNCTION2" | tee -a
 "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "You can view your dashboard in the CloudWatch console:" | tee -a
 "$LOG_FILE"
echo "https://console.aws.amazon.com/cloudwatch/home#dashboards:name=
$DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"

# Auto-confirm cleanup
```

```
echo "" | tee -a "$LOG_FILE"
echo "========================================" | tee -a "$LOG_FILE"
echo "CLEANUP CONFIRMATION" | tee -a "$LOG_FILE"
echo "========================================" | tee -a "$LOG_FILE"
echo "The following resources were created:" | tee -a "$LOG_FILE"
echo "- CloudWatch Dashboard: $DASHBOARD_NAME" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION1" | tee -a "$LOG_FILE"
echo "- Lambda Function: $LAMBDA_FUNCTION2" | tee -a "$LOG_FILE"
echo "- IAM Role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"
echo "Auto-confirming cleanup of all created resources..." | tee -a "$LOG_FILE"

echo "Script completed successfully." | tee -a "$LOG_FILE"
exit 0
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetDashboard
  - Invocar
  - PutDashboard

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Trabalhar com fluxos e vida útil do DynamoDB usando a AWS Command Line Interface v2

O exemplo de código apresentado a seguir demonstra como gerenciar os recursos de fluxos e de vida útil do DynamoDB.

- Crie uma tabela com a opção de fluxos habilitada.
- Descreva os fluxos.
- Crie uma função do Lambda para processar os fluxos.
- Habilite a vida útil (TTL) em uma tabela.
- Adicione itens com atributos de TTL.
- Descreva as configurações de TTL.

Bash

AWS CLI com script Bash

Crie uma tabela com a opção de fluxos habilitada.

```
# Create a table with DynamoDB Streams enabled
aws dynamodb create-table \
    --table-name StreamsDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST \
    --stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

Descreva os fluxos.

```
# Get information about the stream
aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.StreamSpecification"

# Get the stream ARN
STREAM_ARN=$(aws dynamodb describe-table \
    --table-name StreamsDemo \
    --query "Table.LatestStreamArn" \
    --output text)

echo "Stream ARN: $STREAM_ARN"

# Describe the stream
```

```
aws dynamodbstreams describe-stream \
    --stream-arn $STREAM_ARN
```

Crie uma função do Lambda para os fluxos.

```
# Step 1: Create an IAM role for the Lambda function
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

aws iam create-role \
    --role-name DynamoDBStreamsLambdaRole \
    --assume-role-policy-document file://trust-policy.json

# Step 2: Attach permissions to the role
aws iam attach-role-policy \
    --role-name DynamoDBStreamsLambdaRole \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AWSLambdaDynamoDBExecutionRole

# Step 3: Create a Lambda function (code would be in a separate file)
echo "Lambda function creation would be done separately with appropriate code"

# Step 4: Create an event source mapping
echo "Example command to create event source mapping:"
echo "aws lambda create-event-source-mapping \\"
echo "    --function-name ProcessDynamoDBRecords \\"
echo "    --event-source $STREAM_ARN \\"
echo "    --batch-size 100 \\"
echo "    --starting-position LATEST"
```

Habilite a vida útil (TTL) em uma tabela.

```
# Create a table for TTL demonstration
aws dynamodb create-table \
    --table-name TTLDemo \
    --attribute-definitions \
        AttributeName=ID,AttributeType=S \
    --key-schema \
        AttributeName=ID,KeyType=HASH \
    --billing-mode PAY_PER_REQUEST

# Wait for table to become active
aws dynamodb wait table-exists --table-name TTLDemo

# Enable TTL on the table
aws dynamodb update-time-to-live \
    --table-name TTLDemo \
    --time-to-live-specification "Enabled=true, AttributeName=ExpirationTime"
```

Adicione itens com atributos de TTL.

```
# Calculate expiration time (current time + 1 day in seconds)
EXPIRATION_TIME=$(date -d "+1 day" +%s)

# Add an item with TTL attribute
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item1"},
        "Data": {"S": "This item will expire in 1 day"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME'"}
    }'

# Add an item that expires in 1 hour
EXPIRATION_TIME_HOUR=$(date -d "+1 hour" +%s)
aws dynamodb put-item \
    --table-name TTLDemo \
    --item '{
        "ID": {"S": "item2"},
        "Data": {"S": "This item will expire in 1 hour"},
        "ExpirationTime": {"N": "'$EXPIRATION_TIME_HOUR'"}
    }'
```

Descreva as configurações de TTL.

```
# Describe TTL settings for a table
aws dynamodb describe-time-to-live \
    --table-name TTLDemo
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateRole
  - CreateTable
  - DescribeTable
  - DescribeTimeToLive
  - PutItem
  - UpdateTimeToLive

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Trabalhar com a API IAM Policy Builder usando um AWS SDK

O código de exemplo abaixo mostra como:

- Criar políticas do IAM usando a API orientada por objetos.
- Use a API IAM Policy Builder com o serviço do IAM.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Os exemplos usam as importações a seguir.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Crie uma política com base no tempo.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
```

```
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.DATE_GREATER_THAN)

.key("aws:CurrentTime")

.value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.DATE_LESS_THAN)

.key("aws:CurrentTime")

.value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

Crie uma política com várias condições.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

.addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

.addAction("dynamodb:BatchWriteItem")

.addResource("arn:aws:dynamodb:*:*:table/table-name")
```

```
 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                    List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))

 .key("dynamodb:Select")

 .value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Use entidades principais em uma política.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.ARN_NOT_EQUALS)

 .key("aws:PrincipalArn")

 .value("arn:aws:iam::444455556666:user/user-name")))
```

```
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Permitir o acesso entre contas ao .

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)

 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::amzn-
s3-demo-bucket/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                .key("s3:x-amz-
acl")
                                                                .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Crie e carregue uma `IamPolicy`.

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
```

```
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

Baixe e trabalhe com uma `IamPolicy`.

```
        public String createNewBasedOnExistingPolicyExample(IamClient iam, String
 accountID, String policyName,
                        String newPolicyName) {

                String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
 policyName;
                GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
 r.policyArn(policyArn));

                String policyVersion =
 getPolicyResponse.policy().defaultVersionId();
                GetPolicyVersionResponse getPolicyVersionResponse = iam
                                .getPolicyVersion(r ->
 r.policyArn(policyArn).versionId(policyVersion));

                // Create an IamPolicy instance from the JSON string returned
 from IAM.
                String decodedPolicy =
 URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                                StandardCharsets.UTF_8);
                IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

                /*
                 * All IamPolicy components are immutable, so use the copy method
 that creates a
                 * new instance that
                 * can be altered in the same method call.
                 *
```

```
                 * Add the ability to get an item from DynamoDB as an additional
action.
                 */
                IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

                // Create a new statement that replaces the original statement.
                IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

                // Upload the new policy. IAM now has both policies.
                iam.createPolicy(r -> r.policyName(newPolicyName)
                                .policyDocument(newPolicy.toJson()));

                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK for Java 2.x.
- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK for Java 2.x.
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código para o AWS STS usando AWS SDKs

Os exemplos de código a seguir mostram como usar o AWS STS com um kit de desenvolvimento de software (SDK) da AWS.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para ver uma lista completa dos Guias do desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

AWS STS

- Exemplos básicos para o AWS STS usando SDKs da AWS
  - Ações para o AWS STS usando AWS SDKs
    - Usar AssumeRole com o AWS SDK ou a CLI
    - Usar AssumeRoleWithWebIdentity com uma CLI
    - Usar DecodeAuthorizationMessage com uma CLI
    - Usar GetFederationToken com uma CLI
    - Usar GetSessionToken com o AWS SDK ou a CLI
- Cenários para o AWS STS usando AWS SDKs
  - Assumir um perfil do IAM que exija um token de MFA com o AWS STS usando um AWS SDK
  - Configurar a conectividade do serviço de contêiner
  - Crie uma URL com o AWS STS para usuários federados usando um AWS SDK
  - Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
  - Criar uma API REST com integração de proxy de função
  - Como criar um serviço de contêiner para instâncias de máquinas virtuais
  - Como criar um espaço de trabalho de monitoramento gerenciado
  - Como criar um painel de monitoramento com o nome da função como uma variável
  - Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK
  - Conceitos básicos do gerenciamento de configurações
  - Conceitos básicos de registros de contêineres
  - Conceitos básicos de mensagens da Internet das Coisas
  - Conceitos básicos de arquivos de atributos de machine learning
  - Conceitos básicos de clusters gerenciados de kubernetes
  - Conceitos básicos de streaming gerenciado

- Conceitos básicos das notificações push
- Conceitos básicos dos mecanismos de pesquisa e analytics
- Conceitos básicos de firewalls do aplicativo Web
- Como mover segredos codificados para o armazenamento seguro de segredos
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas

Exemplos básicos para o AWS STS usando SDKs da AWS

Os exemplos de código a seguir mostram como usar os conceitos básicos do AWS Security Token Service com SDKs da AWS.

Exemplos

- Ações para o AWS STS usando AWS SDKs
  - Usar AssumeRole com o AWS SDK ou a CLI
  - Usar AssumeRoleWithWebIdentity com uma CLI
  - Usar DecodeAuthorizationMessage com uma CLI
  - Usar GetFederationToken com uma CLI
  - Usar GetSessionToken com o AWS SDK ou a CLI

Ações para o AWS STS usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do AWS STS com AWS SDKs. Cada exemplo inclui um link para o GitHub, em que é possível encontrar instruções para configurar e executar o código.

Esses trechos chamam a API do AWS STS e são trechos de código de programas maiores que devem ser executados no contexto. É possível ver as ações em contexto em Cenários para o AWS STS usando AWS SDKs.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a Referência de APIs do AWS Security Token Service.

Exemplos

- Usar AssumeRole com o AWS SDK ou a CLI
- Usar AssumeRoleWithWebIdentity com uma CLI
- Usar DecodeAuthorizationMessage com uma CLI
- Usar GetFederationToken com uma CLI
- Usar GetSessionToken com o AWS SDK ou a CLI

Usar **`AssumeRole`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `AssumeRole`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto nos seguintes exemplos de código:

- Assumir um perfil do IAM que exija um token de MFA
- Criar um URL para usuários federados usando

.NET

SDK para .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
```

```
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        // A region property may be used if the profile or credentials loaded do
 not specify a region,
        // or to use a specific region.
        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);
```

```
            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials, REGION);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Consulte detalhes da API em AssumeRole na Referência de comandos da AWS CLI.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
```

```
                            Aws::Auth::AWSCredentials &credentials,
                            const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para C++.

CLI

AWS CLI

Como assumir um perfil

O comando `assume-role`, apresentado a seguir, recupera um conjunto de credenciais de curto prazo para o perfil do IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Resultado:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

A saída do comando contém uma chave de acesso, uma chave secreta e um token de sessão que você pode usar para se autenticar na AWS.

Para o uso da AWS CLI, é possível configurar um perfil nomeado associado a um perfil. Ao usar o perfil, a AWS CLI chamará assume-role e gerenciará credenciais para você. Para obter mais informações, consulte Uso de perfis do IAM na AWS CLI no Guia do usuário da AWS CLI.

- Consulte detalhes da API em AssumeRole na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version":"2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
```

```
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, arn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
```

```
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no Repositório de exemplos de código da AWS.

Crie o cliente.

```
import { STSClient } from "@aws-sdk/client-sts";
```

```
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Assuma um perfil do IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);
```

```
  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para JavaScript.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: retorna um conjunto de credenciais temporárias (chave de acesso, chave secreta e token de sessão) que, durante uma hora, podem ser usadas para acessar recursos da AWS aos quais o usuário solicitante normalmente não teria acesso. As credenciais retornadas têm as permissões permitidas pela política de acesso do perfil assumido e pela política fornecida (não é possível usar a política fornecida para conceder permissões além das definidas pela política de acesso do perfil que está sendo assumido).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemplo 2: retorna um conjunto de credenciais temporárias, válidas por uma hora, que têm as mesmas permissões definidas na política de acesso do perfil que está sendo assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemplo 3: retorna um conjunto de credenciais temporárias que fornecem o número de série e o token gerado de uma MFA associada às credenciais do usuário usadas para executar o cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemplo 4: retorna um conjunto de credenciais temporárias que assumiram um perfil definido em uma conta de cliente. Para cada perfil que o terceiro possa assumir, a conta do cliente deve criar um perfil usando um identificador a ser transmitido no parâmetro -ExternalId sempre que o perfil for assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Consulte detalhes da API em AssumeRole na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: retorna um conjunto de credenciais temporárias (chave de acesso, chave secreta e token de sessão) que, durante uma hora, podem ser usadas para acessar recursos da AWS aos quais o usuário solicitante normalmente não teria acesso. As credenciais retornadas têm as permissões permitidas pela política de acesso do perfil assumido e pela política fornecida (não é possível usar a política fornecida para conceder permissões além das definidas pela política de acesso do perfil que está sendo assumido).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemplo 2: retorna um conjunto de credenciais temporárias, válidas por uma hora, que têm as mesmas permissões definidas na política de acesso do perfil que está sendo assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemplo 3: retorna um conjunto de credenciais temporárias que fornecem o número de série e o token gerado de uma MFA associada às credenciais do usuário usadas para executar o cmdlet.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemplo 4: retorna um conjunto de credenciais temporárias que assumiram um perfil definido em uma conta de cliente. Para cada perfil que o terceiro possa assumir, a conta do cliente deve criar um perfil usando um identificador a ser transmitido no parâmetro -ExternalId sempre que o perfil for assumido.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Para obter detalhes sobre a API, consulte AssumeRole na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Assuma um perfil do IAM que exija um token de MFA e use credenciais temporárias para listar os buckets do Amazon S3 para a conta.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
```

```
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
```

```
  # @param key_secret [String] The secret part of the access key used by the STS
client.
  def create_sts_client(key_id, key_secret)
    Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
  end

  # Gets temporary credentials that can be used to assume a role.
  #
  # @param role_arn [String] The ARN of the role that is assumed when these
credentials
  #                          are used.
  # @param sts_client [AWS::STS::Client] An AWS STS client.
  # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
  def assume_role(role_arn, sts_client)
    credentials = Aws::AssumeRoleCredentials.new(
      client: sts_client,
      role_arn: role_arn,
      role_session_name: 'create-use-assume-role-scenario'
    )
    @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
    credentials
  end
```

- Consulte detalhes da API em AssumeRole na Referência da API AWS SDK para Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
```

```
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Consulte detalhes da API em AssumeRole na Referência do AWS SDK para API Rust.

Swift

SDK para Swift

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
import AWSSTS

    public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
        async throws -> STSClientTypes.Credentials
    {
        let input = AssumeRoleInput(
            roleArn: role.arn,
```

```
            roleSessionName: sessionName
        )
        do {
            let output = try await stsClient.assumeRole(input: input)

            guard let credentials = output.credentials else {
                throw ServiceHandlerError.authError
            }

            return credentials
        } catch {
            print("Error assuming role: ", dump(error))
            throw error
        }
    }
```

- Para obter detalhes da API, consulte AssumeRole na Referência do AWS SDK para API Swift.

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AssumeRoleWithWebIdentity`** com uma CLI

Os exemplos de código a seguir mostram como usar o `AssumeRoleWithWebIdentity`.

CLI

AWS CLI

Obter credenciais de curto prazo para um perfil autenticado com identidade Web (OAuth 2."0)

O comando `assume-role-with-web-identity`, apresentado a seguir, recupera um conjunto de credenciais de curto prazo para o perfil do IAM `app1`. A solicitação é autenticada com o token de identidade Web fornecido pelo provedor de identidade Web especificado. Duas políticas adicionais são aplicadas à sessão para restringir ainda mais o que o usuário pode fazer. As credenciais retornadas expiram uma hora após serem geradas.

```
aws sts assume-role-with-web-identity \
```

```
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

Resultado:

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A",
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    },
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

Para obter mais informações, consulte Solicitação de credenciais de segurança temporárias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em AssumeRoleWithWebIdentity na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: retorna um conjunto temporário de credenciais, válido por uma hora, para um usuário que foi autenticado com o provedor de identidade Login with Amazon. As credenciais assumem a política de acesso associada ao perfil identificado pelo ARN do perfil. Opcionalmente, você pode transmitir uma política JSON ao parâmetro -Policy que refina ainda mais as permissões de acesso (não é possível conceder mais permissões do que as disponíveis nas permissões associadas ao perfil). O valor fornecido ao -WebIdentityToken é o identificador de usuário exclusivo que foi retornado pelo provedor de identidade.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Consulte detalhes da API em AssumeRoleWithWebIdentity na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: retorna um conjunto temporário de credenciais, válido por uma hora, para um usuário que foi autenticado com o provedor de identidade Login with Amazon. As credenciais assumem a política de acesso associada ao perfil identificado pelo ARN do perfil. Opcionalmente, você pode transmitir uma política JSON ao parâmetro -Policy que refina ainda mais as permissões de acesso (não é possível conceder mais permissões do que as disponíveis nas permissões associadas ao perfil). O valor fornecido ao -WebIdentityToken é o identificador de usuário exclusivo que foi retornado pelo provedor de identidade.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Para obter detalhes sobre a API, consulte AssumeRoleWithWebIdentity na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **DecodeAuthorizationMessage** com uma CLI

Os exemplos de código a seguir mostram como usar o `DecodeAuthorizationMessage`.

CLI

AWS CLI

Para decodificar uma mensagem de autorização codificada retornada em resposta a uma solicitação

O exemplo `decode-authorization-message` a seguir decodifica informações adicionais sobre o status da autorização de uma solicitação de uma mensagem codificada retornada em resposta a uma solicitação da Amazon Web Services.

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

Resultado:

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
```

```
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```

Para obter mais informações, consulte Lógica da avaliação de política no Guia do usuário do AWS IAM.

- Consulte detalhes da API em DecodeAuthorizationMessage na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: decodifica as informações adicionais contidas no conteúdo da mensagem codificada fornecida que foi retornada em resposta a uma solicitação. As informações adicionais são codificadas porque os detalhes do status da autorização podem constituir informações privilegiadas que o usuário responsável por solicitar a ação não deve ver.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Consulte detalhes da API em DecodeAuthorizationMessage na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: decodifica as informações adicionais contidas no conteúdo da mensagem codificada fornecida que foi retornada em resposta a uma solicitação. As informações adicionais são codificadas porque os detalhes do status da autorização podem constituir informações privilegiadas que o usuário responsável por solicitar a ação não deve ver.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Para obter detalhes sobre a API, consulte DecodeAuthorizationMessage na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetFederationToken`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetFederationToken`.

CLI

AWS CLI

Para retornar um conjunto de credenciais de segurança temporárias usando as credenciais da chave de acesso do usuário do IAM

O exemplo `get-federation-token` a seguir retorna um conjunto de credenciais de segurança temporárias (que consistem em um ID de chave de acesso, uma chave de acesso secreta e um token de segurança) para um usuário. Você deve chamar a operação `GetFederationToken` usando as credenciais de segurança de longo prazo de um usuário do IAM.

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

Conteúdo de `myfile.json`:

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

Resultado:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
    "PackedPolicySize": 36
}
```

Para obter mais informações, consulte Solicitação de credenciais de segurança temporárias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetFederationToken na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: solicita um token federado válido por uma hora usando "Bob" como nome do usuário federado. Esse nome pode ser usado para referenciar o nome do usuário federado em uma política baseada em recursos (como uma política de bucket do Amazon S3). A política do IAM fornecida, no formato JSON, é usada para definir o escopo das permissões que estão disponíveis para o usuário do IAM. A política fornecida não pode conceder mais permissões do que as concedidas ao usuário solicitante, com as permissões finais do usuário federado sendo o conjunto mais restritivo com base na interseção da política aprovada com a política de usuário do IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Para ver detalhes da API, consulte GetFederationToken na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: solicita um token federado válido por uma hora usando "Bob" como nome do usuário federado. Esse nome pode ser usado para referenciar o nome do usuário federado em uma política baseada em recursos (como uma política de bucket do Amazon S3). A política do IAM fornecida, no formato JSON, é usada para definir o escopo das permissões que estão disponíveis para o usuário do IAM. A política fornecida não pode conceder mais permissões do que as concedidas ao usuário solicitante, com as permissões finais do usuário federado sendo o conjunto mais restritivo com base na interseção da política aprovada com a política de usuário do IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Para obter detalhes sobre a API, consulte GetFederationToken na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetSessionToken`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetSessionToken`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Obtenha um token de sessão que requeira um token de MFA

CLI

AWS CLI

Como obter um conjunto de credenciais de curto prazo para uma identidade do IAM

O comando `get-session-token`, apresentado a seguir, recupera um conjunto de credenciais de curto prazo para a identidade do IAM que executa a chamada. As credenciais resultantes podem ser usadas para solicitações em que a autenticação multifator (MFA) é requerida pela política. As credenciais expiram 15 minutos após serem geradas.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Resultado:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
```

```
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Para obter mais informações, consulte Solicitação de credenciais de segurança temporárias no Guia do usuário do AWS IAM.

- Consulte detalhes da API em GetSessionToken na Referência de comandos da AWS CLI.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por um determinado período. As credenciais usadas para solicitar credenciais temporárias são inferidas dos padrões atuais do shell. Para especificar outras credenciais, use os parâmetros -ProfileName ou -AccessKey/-SecretKey.

```
Get-STSSessionToken
```

Saída:

```
AccessKeyId                          Expiration
 SecretAccessKey                     SessionToken
-----------                          ----------
 ---------------                     ------------
EXAMPLEACCESSKEYID                   2/16/2015 9:12:28 PM
 examplesecretaccesskey...           SamPleTokeN.....
```

Exemplo 2: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora. As credenciais usadas para fazer a solicitação são obtidas do perfil especificado.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Saída:

```
AccessKeyId                          Expiration
 SecretAccessKey                     SessionToken
```

```
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemplo 3: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora usando o número de identificação do dispositivo de MFA associado à conta cujas credenciais estão especificadas no perfil 'myprofilename' e o valor fornecido pelo dispositivo.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Consulte detalhes da API em GetSessionToken na Referência de cmdlets do Ferramentas da AWS para PowerShell (V4).

Ferramentas para o PowerShell V5

Exemplo 1: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por um determinado período. As credenciais usadas para solicitar credenciais temporárias são inferidas dos padrões atuais do shell. Para especificar outras credenciais, use os parâmetros -ProfileName ou -AccessKey/-SecretKey.

```
Get-STSSessionToken
```

Saída:

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
```

```
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

Exemplo 2: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora. As credenciais usadas para fazer a solicitação são obtidas do perfil especificado.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Saída:

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

Exemplo 3: retorna uma instância **`Amazon.RuntimeAWSCredentials`** contendo credenciais temporárias válidas por uma hora usando o número de identificação do dispositivo de MFA associado à conta cujas credenciais estão especificadas no perfil 'myprofilename' e o valor fornecido pelo dispositivo.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Saída:

```
AccessKeyId                             Expiration
 SecretAccessKey                        SessionToken
-----------                             ----------
 ---------------                        ------------
EXAMPLEACCESSKEYID                      2/16/2015 9:12:28 PM
 examplesecretaccesskey...              SamPleTokeN.....
```

- Para obter detalhes sobre a API, consulte GetSessionToken na Ferramentas da AWS para PowerShell Cmdlet Reference (V5).

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Obtenha um token de sessão passando um token de MFA e use-o para listar os buckets do Amazon S3 para a conta.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
```

```
    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Consulte detalhes da API em GetSessionToken na Referência da API AWS SDK for Python (Boto3).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cenários para o AWS STS usando AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns no AWS STS com AWS SDKs. Esses cenários mostram como realizar tarefas específicas chamando várias funções no AWS STS ou combinadas com outros Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, em que você pode encontrar instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudar você a compreender ações de serviço em contexto.

Exemplos

- Assumir um perfil do IAM que exija um token de MFA com o AWS STS usando um AWS SDK
- Configurar a conectividade do serviço de contêiner
- Crie uma URL com o AWS STS para usuários federados usando um AWS SDK
- Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor
- Criar uma API REST com integração de proxy de função
- Como criar um serviço de contêiner para instâncias de máquinas virtuais
- Como criar um espaço de trabalho de monitoramento gerenciado
- Como criar um painel de monitoramento com o nome da função como uma variável
- Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK
- Conceitos básicos do gerenciamento de configurações

- Conceitos básicos de registros de contêineres
- Conceitos básicos de mensagens da Internet das Coisas
- Conceitos básicos de arquivos de atributos de machine learning
- Conceitos básicos de clusters gerenciados de kubernetes
- Conceitos básicos de streaming gerenciado
- Conceitos básicos das notificações push
- Conceitos básicos dos mecanismos de pesquisa e analytics
- Conceitos básicos de firewalls do aplicativo Web
- Como mover segredos codificados para o armazenamento seguro de segredos
- Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas
- Como configurar o gerenciamento de sistemas

Assumir um perfil do IAM que exija um token de MFA com o AWS STS usando um AWS SDK

O exemplo de código a seguir mostra como assumir um perfil que exige um token de MFA.

> **Warning**
>
> Para evitar riscos de segurança, não use usuários do IAM para autenticação ao desenvolver software com propósito específico ou trabalhar com dados reais. Em vez disso, use federação com um provedor de identidade, como Centro de Identidade do AWS IAM.

- Criar um perfil do IAM que conceda permissão para listar os buckets do Amazon S3.
- Criar um usuário do IAM que tenha permissão para assumir o perfil somente quando as credenciais de MFA forem fornecidas.
- Registrar um dispositivo MFA para o usuário.
- Assumir o perfil e usar credenciais temporárias para listar buckets do S3.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário do IAM, registre um dispositivo de MFA e crie um perfil que conceda permissão para listar os buckets do Amazon S3. O usuário só tem direitos para assumir a função.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
 MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    For demonstration purposes, the user is created in the same account as the
 role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
```

```
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
```

```
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
```

```
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Mostre que não é permitido assumir uma função sem um token de MFA.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
 results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
 RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

Assuma o perfil que concede permissão para listar os buckets do S3 passando o token de MFA necessário e mostre que os buckets podem ser listados.

```
def list_buckets_from_assumed_role_with_mfa(
```

```
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Destrua os recursos criados para a demonstração.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Execute esse cenário usando as funções definidas anteriormente.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
```

```
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for Python (Boto3).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Configurar a conectividade do serviço de contêiner

O código de exemplo abaixo mostra como:

- Criar a infraestrutura da VPC.
- Configurar registro em log
- Criar o cluster do ECS.
- Configurar perfis do IAM.
- Criar o serviço com o Service Connect.
- Verificar a implantação
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# ECS Service Connect Tutorial Script v4 - Modified to use Default VPC
# This script creates an ECS cluster with Service Connect and deploys an nginx
 service
# Uses the default VPC to avoid VPC limits

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ECS Service Connect Tutorial"
LOG_FILE="ecs-service-connect-tutorial-v4-default-vpc.log"
REGION=${AWS_DEFAULT_REGION:-${AWS_REGION:-$(aws configure get region 2>/dev/
null)}}
if [ -z "$REGION" ]; then
    echo "ERROR: No AWS region configured."
    echo "Set one with: aws configure set region us-east-1"
    exit 1
fi
ENV_PREFIX="tutorial"
CLUSTER_NAME="${ENV_PREFIX}-cluster"
NAMESPACE_NAME="service-connect"

# Generate random suffix for unique resource names
RANDOM_SUFFIX=$(openssl rand -hex 6)

# Arrays to track created resources for cleanup
declare -a CREATED_RESOURCES=()

# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
```

```
}

# Error handling function
handle_error() {
    log "ERROR: Script failed at line $1"
    log "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set up error handling
trap 'handle_error $LINENO' ERR

# Function to add resource to tracking array
track_resource() {
    CREATED_RESOURCES+=("$1")
    log "Tracking resource: $1"
}

# Function to check if command output contains actual errors
check_for_errors() {
    local output="$1"
    local command_name="$2"

    # Check for specific AWS CLI error patterns, not just any occurrence of
 "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameterException\|
AccessDenied\|ValidationException\|ResourceNotFoundException"; then
        log "ERROR in $command_name: $output"
        return 1
    fi
    return 0
}

# Function to get AWS account ID
get_account_id() {
    ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
    log "Using AWS Account ID: $ACCOUNT_ID"
}

# Function to wait for resources to be ready
wait_for_resource() {
    local resource_type="$1"
    local resource_id="$2"
```

```
    case "$resource_type" in
        "cluster")
            log "Waiting for cluster $resource_id to be active..."
            local attempt=1
            local max_attempts=30
            while [ $attempt -le $max_attempts ]; do
                local status=$(aws ecs describe-clusters --clusters
 "$resource_id" --query 'clusters[0].status' --output text)
                if [ "$status" = "ACTIVE" ]; then
                    log "Cluster is now active"
                    return 0
                fi
                log "Cluster status: $status (attempt $attempt/$max_attempts)"
                sleep 10
                ((attempt++))
            done
            log "ERROR: Cluster did not become active within expected time"
            return 1
            ;;
        "service")
            log "Waiting for service $resource_id to be stable..."
            aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id"
            ;;
        "nat-gateway")
            log "Waiting for NAT Gateway $resource_id to be available..."
            aws ec2 wait nat-gateway-available --nat-gateway-ids "$resource_id"
            ;;
    esac
}

# Function to use default VPC infrastructure
setup_default_vpc_infrastructure() {
    log "Using default VPC infrastructure..."

    # Get default VPC
    VPC_ID=$(aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query
 'Vpcs[0].VpcId' --output text)
    if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
        log "ERROR: No default VPC found. Please create a default VPC first."
        exit 1
    fi
    log "Using default VPC: $VPC_ID"
```

```
    # Get default subnets
    SUBNETS=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 "Name=default-for-az,Values=true" --query 'Subnets[].SubnetId' --output text)
    SUBNET_ARRAY=($SUBNETS)

    if [ ${#SUBNET_ARRAY[@]} -lt 2 ]; then
        log "ERROR: Need at least 2 subnets for ECS Service Connect. Found:
 ${#SUBNET_ARRAY[@]}"
        exit 1
    fi

    PUBLIC_SUBNET1=${SUBNET_ARRAY[0]}
    PUBLIC_SUBNET2=${SUBNET_ARRAY[1]}

    log "Using subnets: $PUBLIC_SUBNET1, $PUBLIC_SUBNET2"

    # Create security group for ECS tasks
    SG_OUTPUT=$(aws ec2 create-security-group \
        --group-name "${ENV_PREFIX}-ecs-sg-${RANDOM_SUFFIX}" \
        --description "Security group for ECS Service Connect tutorial" \
        --vpc-id "$VPC_ID" \
        --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-ecs-service-
connect}]' 2>&1)
    check_for_errors "$SG_OUTPUT" "create-security-group"
    SECURITY_GROUP_ID=$(echo "$SG_OUTPUT" | grep -o '"GroupId": "[^"]*"' | cut -
d'"' -f4)
    track_resource "SG:$SECURITY_GROUP_ID"
    log "Created security group: $SECURITY_GROUP_ID"

    # Add inbound rules to security group
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true

    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 443 \
        --cidr 0.0.0.0/0 >/dev/null 2>&1 || true
```

```
    log "Default VPC infrastructure setup completed"
}

# Function to create CloudWatch log groups
create_log_groups() {
    log "Creating CloudWatch log groups..."

    # Create log group for nginx container
    aws logs create-log-group --log-group-name "/ecs/service-connect-nginx"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-nginx created"
            track_resource "LOG_GROUP:/ecs/service-connect-nginx"
        else
            log "Log group /ecs/service-connect-nginx already exists"
        fi
    }

    # Create log group for service connect proxy
    aws logs create-log-group --log-group-name "/ecs/service-connect-proxy"
 --tags project=doc-smith,tutorial=amazon-ecs-service-connect 2>&1 | grep -v
 "ResourceAlreadyExistsException" || {
        if [ ${PIPESTATUS[0]} -eq 0 ]; then
            log "Log group /ecs/service-connect-proxy created"
            track_resource "LOG_GROUP:/ecs/service-connect-proxy"
        else
            log "Log group /ecs/service-connect-proxy already exists"
        fi
    }
}

# Function to create ECS cluster with Service Connect
create_ecs_cluster() {
    log "Creating ECS cluster with Service Connect..."

    CLUSTER_OUTPUT=$(aws ecs create-cluster \
        --cluster-name "$CLUSTER_NAME" \
        --service-connect-defaults namespace="$NAMESPACE_NAME" \
        --tags key=Environment,value=tutorial key=project,value=doc-smith
 key=tutorial,value=amazon-ecs-service-connect 2>&1)
    check_for_errors "$CLUSTER_OUTPUT" "create-cluster"

    track_resource "CLUSTER:$CLUSTER_NAME"
```

```
    log "Created ECS cluster: $CLUSTER_NAME"

    wait_for_resource "cluster" "$CLUSTER_NAME"

    # Track the Service Connect namespace that gets created
    # Wait a moment for the namespace to be created
    sleep 5
    NAMESPACE_ID=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_ID" && "$NAMESPACE_ID" != "None" ]]; then
        track_resource "NAMESPACE:$NAMESPACE_ID"
        log "Service Connect namespace created: $NAMESPACE_ID"
    fi
}

# Function to create IAM roles
create_iam_roles() {
    log "Creating IAM roles..."

    # Check if ecsTaskExecutionRole exists
    if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
        log "IAM role ecsTaskExecutionRole exists"
    else
        log "Creating ecsTaskExecutionRole..."
        aws iam create-role \
            --role-name ecsTaskExecutionRole \
            --assume-role-policy-document '{
                "Version":"2012-10-17",
                "Statement": [{
                    "Effect": "Allow",
                    "Principal": {"Service": "ecs-tasks.amazonaws.com"},
                    "Action": "sts:AssumeRole"
                }]
            }' >/dev/null 2>&1
        aws iam attach-role-policy \
            --role-name ecsTaskExecutionRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy >/dev/null 2>&1
        track_resource "ROLE:ecsTaskExecutionRole"
        aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-service-connect
```

```
        log "Created ecsTaskExecutionRole"
        sleep 10
    fi

    # Check if ecsTaskRole exists, create if not
    if aws iam get-role --role-name ecsTaskRole >/dev/null 2>&1; then
        log "IAM role ecsTaskRole exists"
    else
        log "IAM role ecsTaskRole does not exist, will create it"

        # Create trust policy for ECS tasks
        cat > /tmp/ecs-task-trust-policy.json << EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

        aws iam create-role \
            --role-name ecsTaskRole \
            --assume-role-policy-document file:///tmp/ecs-task-trust-policy.json
 >/dev/null

        track_resource "IAM_ROLE:ecsTaskRole"
        aws iam tag-role --role-name ecsTaskRole --tags Key=project,Value=doc-
smith Key=tutorial,Value=amazon-ecs-service-connect
        log "Created ecsTaskRole"

        # Wait for role to be available
        sleep 10
    fi
}

# Function to create task definition
create_task_definition() {
    log "Creating task definition..."
```

```
    # Create task definition JSON
    cat > /tmp/task-definition.json << EOF
{
    "family": "service-connect-nginx",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskRole",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "protocol": "tcp",
                    "name": "nginx-port"
                }
            ],
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/service-connect-nginx",
                    "awslogs-region": "${REGION}",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

    TASK_DEF_OUTPUT=$(aws ecs register-task-definition --cli-input-json file:///
tmp/task-definition.json 2>&1)
    check_for_errors "$TASK_DEF_OUTPUT" "register-task-definition"

    TASK_DEF_ARN=$(echo "$TASK_DEF_OUTPUT" | grep -o '"taskDefinitionArn":
 "[^"]*"' | cut -d'"' -f4)
    track_resource "TASK_DEF:service-connect-nginx"
    log "Created task definition: $TASK_DEF_ARN"
```

```
    # Clean up temporary file
    rm -f /tmp/task-definition.json
}

# Function to create ECS service with Service Connect
create_ecs_service() {
    log "Creating ECS service with Service Connect..."

    # Create service definition JSON
    cat > /tmp/service-definition.json << EOF
{
    "serviceName": "service-connect-nginx-service",
    "cluster": "${CLUSTER_NAME}",
    "taskDefinition": "service-connect-nginx",
    "desiredCount": 1,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": ["${PUBLIC_SUBNET1}", "${PUBLIC_SUBNET2}"],
            "securityGroups": ["${SECURITY_GROUP_ID}"],
            "assignPublicIp": "ENABLED"
        }
    },
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "${NAMESPACE_NAME}",
        "services": [
            {
                "portName": "nginx-port",
                "discoveryName": "nginx",
                "clientAliases": [
                    {
                        "port": 80,
                        "dnsName": "nginx"
                    }
                ]
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "${REGION}",
```

```
                "awslogs-stream-prefix": "ecs-service-connect"
            }
        }
    },
    "tags": [
        {
            "key": "Environment",
            "value": "tutorial"
        },
        {
            "key": "project",
            "value": "doc-smith"
        },
        {
            "key": "tutorial",
            "value": "amazon-ecs-service-connect"
        }
    ]
}
EOF

    SERVICE_OUTPUT=$(aws ecs create-service --cli-input-json file:///tmp/service-
definition.json 2>&1)
    check_for_errors "$SERVICE_OUTPUT" "create-service"

    track_resource "SERVICE:service-connect-nginx-service"
    log "Created ECS service: service-connect-nginx-service"

    wait_for_resource "service" "service-connect-nginx-service"

    # Clean up temporary file
    rm -f /tmp/service-definition.json
}

# Function to verify deployment
verify_deployment() {
    log "Verifying deployment..."

    # Check service status
    SERVICE_STATUS=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].status' --output text)
    log "Service status: $SERVICE_STATUS"
```

```
    # Check running tasks
    RUNNING_COUNT=$(aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].runningCount' --output text)
    log "Running tasks: $RUNNING_COUNT"

    # Get task ARN
    TASK_ARN=$(aws ecs list-tasks \
        --cluster "$CLUSTER_NAME" \
        --service-name "service-connect-nginx-service" \
        --query 'taskArns[0]' --output text)

    if [[ "$TASK_ARN" != "None" && -n "$TASK_ARN" ]]; then
        log "Task ARN: $TASK_ARN"

        # Try to get task IP address
        TASK_IP=$(aws ecs describe-tasks \
            --cluster "$CLUSTER_NAME" \
            --tasks "$TASK_ARN" \
            --query 'tasks[0].attachments[0].details[?
name==`privateIPv4Address`].value' \
            --output text 2>/dev/null || echo "")

        if [[ -n "$TASK_IP" && "$TASK_IP" != "None" ]]; then
            log "Task IP address: $TASK_IP"
        else
            log "Could not retrieve task IP address"
        fi
    fi

    # Check Service Connect namespace
    NAMESPACE_STATUS=$(aws servicediscovery list-namespaces \
        --filters Name=TYPE,Values=HTTP \
        --query "Namespaces[?Name=='$NAMESPACE_NAME'].Id" --output text 2>/dev/
null || echo "")

    if [[ -n "$NAMESPACE_STATUS" && "$NAMESPACE_STATUS" != "None" ]]; then
        log "Service Connect namespace '$NAMESPACE_NAME' is active"
    else
        log "Service Connect namespace '$NAMESPACE_NAME' not found or not active"
    fi
```

```
    # Display Service Connect configuration
    log "Service Connect configuration:"
    aws ecs describe-services \
        --cluster "$CLUSTER_NAME" \
        --services "service-connect-nginx-service" \
        --query 'services[0].serviceConnectConfiguration' 2>/dev/null || true
}

# Function to display created resources
display_resources() {
    echo ""
    echo "==========================================="
    echo "CREATED RESOURCES"
    echo "==========================================="
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "- $resource"
    done
    echo "==========================================="
    echo ""
}

# Function to cleanup resources
cleanup_resources() {
    log "Starting cleanup process..."

    # Delete resources in reverse order of creation
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[i]}"
        resource_type=$(echo "$resource" | cut -d':' -f1)
        resource_id=$(echo "$resource" | cut -d':' -f2)

        log "Cleaning up $resource_type: $resource_id"

        case "$resource_type" in
            "SERVICE")
                aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --desired-count 0 2>&1 | grep -qi "error" && log "Warning: Failed
 to scale down service $resource_id"
                aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$resource_id" 2>/dev/null || true
                aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$resource_id" --force 2>&1 | grep -qi "error" && log "Warning: Failed to delete
 service $resource_id"
                ;;
```

```
            "TASK_DEF")
                TASK_DEF_ARNS=$(aws ecs list-task-definitions --family-prefix
 "$resource_id" --query 'taskDefinitionArns' --output text 2>/dev/null)
                for arn in $TASK_DEF_ARNS; do
                    aws ecs deregister-task-definition --task-definition "$arn"
 >/dev/null 2>&1 || true
                done
                ;;
            "ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "IAM_ROLE")
                aws iam detach-role-policy --role-name "$resource_id" --policy-
arn "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy" 2>/
dev/null || true
                aws iam delete-role --role-name "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete role $resource_id"
                ;;
            "CLUSTER")
                aws ecs delete-cluster --cluster "$resource_id" 2>&1 | grep -qi
 "error" && log "Warning: Failed to delete cluster $resource_id"
                ;;
            "SG")
                for attempt in 1 2 3 4 5; do
                    if aws ec2 delete-security-group --group-id "$resource_id"
 2>/dev/null; then
                        break
                    fi
                    log "Security group $resource_id still has dependencies,
 retrying in 30s ($attempt/5)..."
                    sleep 30
                done
                ;;
            "LOG_GROUP")
                aws logs delete-log-group --log-group-name "$resource_id" 2>&1 |
 grep -qi "error" && log "Warning: Failed to delete log group $resource_id"
                ;;
            "NAMESPACE")
                # First, delete any services in the namespace
                NAMESPACE_SERVICES=$(aws servicediscovery list-services \
```

```
                    --filters Name=NAMESPACE_ID,Values="$resource_id" \
                    --query 'Services[].Id' --output text 2>/dev/null || echo "")

                if [[ -n "$NAMESPACE_SERVICES" && "$NAMESPACE_SERVICES" !=
 "None" ]]; then
                    for service_id in $NAMESPACE_SERVICES; do
                        aws servicediscovery delete-service --id "$service_id" >/
dev/null 2>&1 || true
                        sleep 2
                    done
                fi

                # Then delete the namespace
                aws servicediscovery delete-namespace --id "$resource_id" >/dev/
null 2>&1 || true
                ;;
        esac

        sleep 2  # Brief pause between deletions
    done

    # Clean up temporary files
    rm -f /tmp/ecs-task-trust-policy.json
    rm -f /tmp/task-definition.json
    rm -f /tmp/service-definition.json

    log "Cleanup completed"
}

# Main execution
main() {
    log "Starting $SCRIPT_NAME v4 (Default VPC)"
    log "Region: $REGION"
    log "Log file: $LOG_FILE"

    # Get AWS account ID
    get_account_id

    # Setup infrastructure using default VPC
    setup_default_vpc_infrastructure

    # Create CloudWatch log groups
    create_log_groups
```

```
    # Create ECS cluster
    create_ecs_cluster

    # Create IAM roles
    create_iam_roles

    # Create task definition
    create_task_definition

    # Create ECS service
    create_ecs_service

    # Verify deployment
    verify_deployment

    log "Tutorial completed successfully!"

    # Display created resources
    display_resources

    # Ask user if they want to clean up
    echo ""
    echo "==========================================="
    echo "CLEANUP CONFIRMATION"
    echo "==========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        cleanup_resources
        log "All resources have been cleaned up"
    else
        log "Resources left intact. You can clean them up later by running the
 cleanup function."
        echo ""
        echo "To clean up resources later, you can use the AWS CLI commands or
 the AWS Management Console."
        echo "Remember to delete resources in the correct order to avoid
 dependency issues."
    fi
}

# Make script executable and run
chmod +x "$0"
```

```
main "$@"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateLogGroup
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteLogGroup
  - DeleteNamespace
  - DeleteRole
  - DeleteSecurityGroup
  - DeleteService (AWS Cloud Map)
  - DeleteService (Amazon ECS)
  - DeregisterTaskDefinition
  - DescribeClusters
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNamespaces
  - ListServices
  - ListTaskDefinitions
  - ListTasks

- RegisterTaskDefinition
- UpdateService
- Aguardar (Amazon EC2)
- Aguardar (Amazon ECS)

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Crie uma URL com o AWS STS para usuários federados usando um AWS SDK

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM que conceda acesso somente leitura aos recursos do Amazon S3 da conta atual.
- Obter um token de segurança do endpoint de federação da AWS.
- Crie um URL que possa ser usado para acessar o console com credenciais federadas.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um perfil que conceda acesso somente leitura aos recursos do S3 da conta atual.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
```

```
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return role
```

Obtenha um token de segurança do endpoint de federação da AWS e crie um URL que possa ser usado para acessar o console com credenciais federadas.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
```

```
       AWS federation endpoint.
   3. Builds a URL that can be used in a browser to navigate to the AWS
federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.

   :param assume_role_arn: The role that specifies the permissions that are
granted.
                           The current user must have permission to assume the
role.
   :param session_name: The name for the STS session.
   :param issuer: The organization that issues the URL.
   :param sts_client: A Boto3 STS instance that can assume the role.
   :return: The federated URL.
   """
   response = sts_client.assume_role(
       RoleArn=assume_role_arn, RoleSessionName=session_name
   )
   temp_credentials = response["Credentials"]
   print(f"Assumed role {assume_role_arn} and got temporary credentials.")

   session_data = {
       "sessionId": temp_credentials["AccessKeyId"],
       "sessionKey": temp_credentials["SecretAccessKey"],
       "sessionToken": temp_credentials["SessionToken"],
   }
   aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

   # Make a request to the AWS federation endpoint to get a sign-in token.
   # The requests.get function URL-encodes the parameters and builds the query
string
   # before making the request.
   response = requests.get(
       aws_federated_signin_endpoint,
       params={
           "Action": "getSigninToken",
           "SessionDuration": str(datetime.timedelta(hours=12).seconds),
           "Session": json.dumps(session_data),
       },
   )
   signin_token = json.loads(response.text)
   print(f"Got a sign-in token from the AWS sign-in federation endpoint.")
```

```
    # Make a federated URL that can be used to sign into the AWS Management
 Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Destrua os recursos criados para a demonstração.

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Execute esse cenário usando as funções definidas anteriormente.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
```

```
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for Python (Boto3).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar uma tarefa de contêiner para o tipo de inicialização com tecnologia sem servidor

O código de exemplo abaixo mostra como:

- Criar um cluster.
- Criar uma definição de tarefa.
- Criar o serviço.
- Fazer a limpeza.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon ECS Fargate Tutorial Script - Version 5
# This script creates an ECS cluster, task definition, and service using Fargate
 launch type
# Fixed version with proper resource dependency handling during cleanup
# Security improvements applied

set -e  # Exit on any error

# Initialize logging
LOG_FILE="ecs-fargate-tutorial-v5.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon ECS Fargate tutorial at $(date)"
echo "Log file: $LOG_FILE"

# Generate random identifier for unique resource names
RANDOM_ID=$(openssl rand -hex 6)
CLUSTER_NAME="fargate-cluster-${RANDOM_ID}"
SERVICE_NAME="fargate-service-${RANDOM_ID}"
TASK_FAMILY="sample-fargate-${RANDOM_ID}"
SECURITY_GROUP_NAME="ecs-fargate-sg-${RANDOM_ID}"

# Array to track created resources for cleanup
CREATED_RESOURCES=()

# Function to log and execute commands with input validation
execute_command() {
    local cmd="$1"
    local description="$2"
```

```
    # Validate that cmd is not empty
    if [[ -z "$cmd" ]]; then
        echo "ERROR: Command is empty"
        return 1
    fi

    echo ""
    echo "========================================="
    echo "EXECUTING: $description"
    echo "COMMAND: $cmd"
    echo "========================================="

    local output
    local exit_code
    set +e  # Temporarily disable exit on error
    output=$(eval "$cmd" 2>&1)
    exit_code=$?
    set -e  # Re-enable exit on error

    if [[ $exit_code -eq 0 ]]; then
        echo "SUCCESS: $description"
        echo "OUTPUT: $output"
        return 0
    else
        echo "FAILED: $description"
        echo "EXIT CODE: $exit_code"
        echo "OUTPUT: $output"
        return 1
    fi
}

# Function to check for actual AWS API errors in command output
check_for_aws_errors() {
    local output="$1"
    local description="$2"

    # Look for specific AWS error patterns, not just the word "error"
    if echo "$output" | grep -qi "An error occurred\|InvalidParameter\|
AccessDenied\|ResourceNotFound\|ValidationException"; then
        echo "AWS API ERROR detected in output for: $description"
        echo "Output: $output"
        return 1
    fi
    return 0
```

```
}

# Function to safely extract JSON values
safe_json_extract() {
    local json="$1"
    local key="$2"
    local value

    value=$(echo "$json" | grep -o "\"$key\": \"[^\"]*\"" | cut -d'"' -f4 2>/dev/
null || echo "")
    echo "$value"
}

# Function to wait for network interfaces to be cleaned up
wait_for_network_interfaces_cleanup() {
    local security_group_id="$1"
    local max_attempts=30
    local attempt=1

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    echo "Waiting for network interfaces to be cleaned up..."

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Checking for dependent network
 interfaces..."

        # Check if there are any network interfaces still using this security
 group
        local eni_count
        eni_count=$(aws ec2 describe-network-interfaces \
            --filters "Name=group-id,Values=$security_group_id" \
            --query "length(NetworkInterfaces)" \
            --output text 2>/dev/null || echo "0")

        if [[ "$eni_count" == "0" ]]; then
            echo "No network interfaces found using security group
 $security_group_id"
            return 0
        else
```

```
            echo "Found $eni_count network interface(s) still using security
 group $security_group_id"
            echo "Waiting 10 seconds before next check..."
            sleep 10
            ((attempt++))
        fi
    done

    echo "WARNING: Network interfaces may still be attached after $max_attempts
 attempts"
    echo "This is normal and the security group deletion will be retried"
    return 1
}

# Function to retry security group deletion with exponential backoff
retry_security_group_deletion() {
    local security_group_id="$1"
    local max_attempts=10
    local attempt=1
    local wait_time=5

    # Validate security group ID format
    if [[ ! "$security_group_id" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
        echo "ERROR: Invalid security group ID format: $security_group_id"
        return 1
    fi

    while [[ $attempt -le $max_attempts ]]; do
        echo "Attempt $attempt/$max_attempts: Trying to delete security group
 $security_group_id"

        if execute_command "aws ec2 delete-security-group --group-id
 '$security_group_id'" "Delete security group (attempt $attempt)"; then
            echo "Successfully deleted security group $security_group_id"
            return 0
        else
            if [[ $attempt -eq $max_attempts ]]; then
                echo "FAILED: Could not delete security group $security_group_id
 after $max_attempts attempts"
                echo "This may be due to network interfaces that are still being
 cleaned up by AWS"
                echo "You can manually delete it later using: aws ec2 delete-
security-group --group-id $security_group_id"
                return 1
```

```
            else
                echo "Waiting $wait_time seconds before retry..."
                sleep $wait_time
                wait_time=$((wait_time * 2))  # Exponential backoff
                ((attempt++))
            fi
        fi
    done
}

# Function to cleanup resources with proper dependency handling
cleanup_resources() {
    echo ""
    echo "==========================================="
    echo "CLEANUP PROCESS"
    echo "==========================================="
    echo "The following resources were created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "Auto-confirming cleanup of all created resources..."

    CLEANUP_CHOICE="y"

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup process..."

        # Step 1: Scale service to 0 tasks first, then delete service
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Service: $SERVICE_NAME " ]];
 then
            echo ""
            echo "Step 1: Scaling service to 0 tasks..."
            if execute_command "aws ecs update-service --cluster '$CLUSTER_NAME'
 --service '$SERVICE_NAME' --desired-count 0" "Scale service to 0 tasks"; then
                echo "Waiting for service to stabilize after scaling to 0..."
                execute_command "aws ecs wait services-stable --cluster
 '$CLUSTER_NAME' --services '$SERVICE_NAME'" "Wait for service to stabilize"

                echo "Deleting service..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
 --service '$SERVICE_NAME'" "Delete ECS service"
            else
```

```
                echo "WARNING: Failed to scale service. Attempting to delete
anyway..."
                execute_command "aws ecs delete-service --cluster '$CLUSTER_NAME'
--service '$SERVICE_NAME' --force" "Force delete ECS service"
            fi
        fi

        # Step 2: Wait a bit for tasks to fully terminate
        echo ""
        echo "Step 2: Waiting for tasks to fully terminate..."
        sleep 15

        # Step 3: Delete cluster
        if [[ " ${CREATED_RESOURCES[*]} " =~ " ECS Cluster: $CLUSTER_NAME " ]];
then
            echo ""
            echo "Step 3: Deleting cluster..."
            execute_command "aws ecs delete-cluster --cluster '$CLUSTER_NAME'"
"Delete ECS cluster"
        fi

        # Step 4: Wait for network interfaces to be cleaned up, then delete
security group
        if [[ -n "$SECURITY_GROUP_ID" && "$SECURITY_GROUP_ID" != "None" ]]; then
            echo ""
            echo "Step 4: Cleaning up security group..."

            # First, wait for network interfaces to be cleaned up
            wait_for_network_interfaces_cleanup "$SECURITY_GROUP_ID"

            # Then retry security group deletion with backoff
            retry_security_group_deletion "$SECURITY_GROUP_ID"
        fi

        # Step 5: Clean up task definition (deregister all revisions)
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
" ]]; then
            echo ""
            echo "Step 5: Deregistering task definition revisions..."

            # Get all revisions of the task definition
            local revisions
```

```
            revisions=$(aws ecs list-task-definitions --family-prefix
 "$TASK_FAMILY" --query "taskDefinitionArns" --output text 2>/dev/null || echo
 "")

            if [[ -n "$revisions" && "$revisions" != "None" ]]; then
                for revision_arn in $revisions; do
                    echo "Deregistering task definition: $revision_arn"
                    execute_command "aws ecs deregister-task-definition --task-
definition '$revision_arn'" "Deregister task definition $revision_arn" || true
                done
            else
                echo "No task definition revisions found to deregister"
            fi
        fi

        echo ""
        echo "============================================"
        echo "CLEANUP COMPLETED"
        echo "============================================"
        echo "All resources have been cleaned up successfully!"

    else
        echo "Cleanup skipped. Resources remain active."
        echo ""
        echo "To clean up manually later, use the following commands in order:"
        echo "1. Scale service to 0: aws ecs update-service --cluster
 $CLUSTER_NAME --service $SERVICE_NAME --desired-count 0"
        echo "2. Wait for stability: aws ecs wait services-stable --cluster
 $CLUSTER_NAME --services $SERVICE_NAME"
        echo "3. Delete service: aws ecs delete-service --cluster $CLUSTER_NAME
 --service $SERVICE_NAME"
        echo "4. Delete cluster: aws ecs delete-cluster --cluster $CLUSTER_NAME"
        echo "5. Wait 2-3 minutes, then delete security group: aws ec2 delete-
security-group --group-id $SECURITY_GROUP_ID"
        if [[ " ${CREATED_RESOURCES[*]} " =~ " Task Definition: $TASK_FAMILY
 " ]]; then
            echo "6. Deregister task definitions: aws ecs list-task-definitions
 --family-prefix $TASK_FAMILY"
            echo "   Then for each ARN: aws ecs deregister-task-definition --
task-definition <ARN>"
        fi
    fi
}
```

```
# Trap to handle script interruption
trap cleanup_resources EXIT

echo "Using random identifier: $RANDOM_ID"
echo "Cluster name: $CLUSTER_NAME"
echo "Service name: $SERVICE_NAME"
echo "Task family: $TASK_FAMILY"

# Step 1: Ensure ECS task execution role exists
echo ""
echo "============================================="
echo "STEP 1: VERIFY ECS TASK EXECUTION ROLE"
echo "============================================="

ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ ! "$ACCOUNT_ID" =~ ^[0-9]{12}$ ]]; then
    echo "ERROR: Invalid AWS Account ID retrieved: $ACCOUNT_ID"
    exit 1
fi

EXECUTION_ROLE_ARN="arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole"

# Check if role exists
if aws iam get-role --role-name ecsTaskExecutionRole >/dev/null 2>&1; then
    echo "ECS task execution role already exists"
else
    echo "Creating ECS task execution role..."

    # Create trust policy with strict validation
    cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
```

```
    # Validate JSON before using
    if ! jq empty trust-policy.json 2>/dev/null; then
        echo "ERROR: Invalid JSON in trust policy"
        rm -f trust-policy.json
        exit 1
    fi

    execute_command "aws iam create-role --role-name ecsTaskExecutionRole --
assume-role-policy-document file://trust-policy.json" "Create ECS task execution
 role"

    aws iam tag-role --role-name ecsTaskExecutionRole --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-ecs-fargate-linux

    execute_command "aws iam attach-role-policy --role-name
 ecsTaskExecutionRole --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy" "Attach ECS task execution policy"

    # Clean up temporary file securely
    shred -vfz -n 3 trust-policy.json 2>/dev/null || rm -f trust-policy.json

    CREATED_RESOURCES+=("IAM Role: ecsTaskExecutionRole")
fi

# Step 2: Create ECS cluster
echo ""
echo "============================================"
echo "STEP 2: CREATE ECS CLUSTER"
echo "============================================"

CLUSTER_OUTPUT=$(execute_command "aws ecs create-cluster --cluster-name
 '$CLUSTER_NAME' --tags key=project,value=doc-smith key=tutorial,value=amazon-
ecs-fargate-linux" "Create ECS cluster")
check_for_aws_errors "$CLUSTER_OUTPUT" "Create ECS cluster"

CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")

# Step 3: Create task definition
echo ""
echo "============================================"
echo "STEP 3: CREATE TASK DEFINITION"
echo "============================================"

# Create task definition JSON with validated inputs
```

```
cat > task-definition.json << EOF
{
    "family": "$TASK_FAMILY",
    "networkMode": "awsvpc",
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "256",
    "memory": "512",
    "executionRoleArn": "$EXECUTION_ROLE_ARN",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:2.4-alpine",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": ["sh", "-c"],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container
 in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html
 && httpd-foreground\""
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/fargate-sample",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ]
}
EOF

# Validate JSON before using
if ! jq empty task-definition.json 2>/dev/null; then
```

```
    echo "ERROR: Invalid JSON in task definition"
    rm -f task-definition.json
    exit 1
fi

TASK_DEF_OUTPUT=$(execute_command "aws ecs register-task-definition --cli-input-
json file://task-definition.json" "Register task definition")
check_for_aws_errors "$TASK_DEF_OUTPUT" "Register task definition"

# Clean up temporary file securely
shred -vfz -n 3 task-definition.json 2>/dev/null || rm -f task-definition.json

CREATED_RESOURCES+=("Task Definition: $TASK_FAMILY")

# Step 4: Set up networking
echo ""
echo "============================================="
echo "STEP 4: SET UP NETWORKING"
echo "============================================="

# Get default VPC ID
VPC_ID=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true" --query
 "Vpcs[0].VpcId" --output text)
if [[ "$VPC_ID" == "None" || -z "$VPC_ID" ]]; then
    echo "ERROR: No default VPC found. Please create a default VPC or specify a
 custom VPC."
    exit 1
fi

# Validate VPC ID format
if [[ ! "$VPC_ID" =~ ^vpc-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid VPC ID format: $VPC_ID"
    exit 1
fi

echo "Using default VPC: $VPC_ID"

# Create security group with restricted access
# Note: This allows HTTP access from anywhere for demo purposes
# In production, restrict source to specific IP ranges or security groups
SECURITY_GROUP_OUTPUT=$(execute_command "aws ec2 create-security-group --
group-name '$SECURITY_GROUP_NAME' --description 'Security group for ECS
 Fargate tutorial - HTTP access' --vpc-id '$VPC_ID' --tag-specifications
```

```
 'ResourceType=security-group,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-ecs-fargate-linux}]'" "Create security group")
check_for_aws_errors "$SECURITY_GROUP_OUTPUT" "Create security group"

SECURITY_GROUP_ID=$(echo "$SECURITY_GROUP_OUTPUT" | grep -o '"GroupId": "[^"]*"'
 | head -1 | cut -d'"' -f4)
if [[ -z "$SECURITY_GROUP_ID" || "$SECURITY_GROUP_ID" == "None" ]]; then
    SECURITY_GROUP_ID=$(aws ec2 describe-security-groups --group-names
 "$SECURITY_GROUP_NAME" --filters "Name=vpc-id,Values=$VPC_ID" --query
 "SecurityGroups[0].GroupId" --output text)
fi

# Validate security group ID format
if [[ ! "$SECURITY_GROUP_ID" =~ ^sg-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid security group ID format: $SECURITY_GROUP_ID"
    exit 1
fi

echo "Created security group: $SECURITY_GROUP_ID"
CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")

# Add HTTP inbound rule
# WARNING: This allows HTTP access from anywhere (0.0.0.0/0)
# In production environments, restrict this to specific IP ranges
execute_command "aws ec2 authorize-security-group-ingress --group-id
 '$SECURITY_GROUP_ID' --protocol tcp --port 80 --cidr 0.0.0.0/0" "Add HTTP
 inbound rule to security group"

# Get subnet IDs from default VPC
echo "Getting subnet IDs from default VPC..."
SUBNET_IDS_RAW=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$VPC_ID"
 --query "Subnets[*].SubnetId" --output text)
if [[ -z "$SUBNET_IDS_RAW" ]]; then
    echo "ERROR: No subnets found in default VPC"
    exit 1
fi

# Convert to proper comma-separated format, handling both spaces and tabs
SUBNET_IDS_COMMA=$(echo "$SUBNET_IDS_RAW" | tr -s '[:space:]' ',' | sed 's/,$//')
echo "Raw subnet IDs: $SUBNET_IDS_RAW"
echo "Formatted subnet IDs: $SUBNET_IDS_COMMA"

# Validate subnet IDs format
if [[ ! "$SUBNET_IDS_COMMA" =~ ^subnet-[a-z0-9]+(,subnet-[a-z0-9]+)*$ ]]; then
```

```
    echo "ERROR: Invalid subnet ID format: $SUBNET_IDS_COMMA"
    exit 1
fi

# Step 5: Create ECS service
echo ""
echo "=========================================="
echo "STEP 5: CREATE ECS SERVICE"
echo "=========================================="

# Create the service with proper JSON formatting for network configuration
SERVICE_CMD="aws ecs create-service --cluster '$CLUSTER_NAME' --service-name
 '$SERVICE_NAME' --task-definition '$TASK_FAMILY' --desired-count 1 --launch-
type FARGATE --network-configuration '{\"awsvpcConfiguration\":{\"subnets
\":[\"$(echo "$SUBNET_IDS_COMMA" | sed 's/,/","/g')\"],\"securityGroups
\":[\"$SECURITY_GROUP_ID\"],\"assignPublicIp\":\"ENABLED\"}}' --tags
 key=project,value=doc-smith key=tutorial,value=amazon-ecs-fargate-linux"

echo "Service creation command: $SERVICE_CMD"

SERVICE_OUTPUT=$(execute_command "$SERVICE_CMD" "Create ECS service")
check_for_aws_errors "$SERVICE_OUTPUT" "Create ECS service"

CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

# Step 6: Wait for service to stabilize and get public IP
echo ""
echo "=========================================="
echo "STEP 6: WAIT FOR SERVICE AND GET PUBLIC IP"
echo "=========================================="

echo "Waiting for service to stabilize (this may take a few minutes)..."
execute_command "aws ecs wait services-stable --cluster '$CLUSTER_NAME' --
services '$SERVICE_NAME'" "Wait for service to stabilize"

# Get task ARN
TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query "taskArns[0]" --output text)
if [[ "$TASK_ARN" == "None" || -z "$TASK_ARN" ]]; then
    echo "ERROR: No running tasks found for service"
    exit 1
fi

echo "Task ARN: $TASK_ARN"
```

```
# Get network interface ID
ENI_ID=$(aws ecs describe-tasks --cluster "$CLUSTER_NAME" --tasks "$TASK_ARN"
 --query "tasks[0].attachments[0].details[?name=='networkInterfaceId'].value" --
output text)
if [[ "$ENI_ID" == "None" || -z "$ENI_ID" ]]; then
    echo "ERROR: Could not retrieve network interface ID"
    exit 1
fi

# Validate ENI ID format
if [[ ! "$ENI_ID" =~ ^eni-[a-z0-9]{8,17}$ ]]; then
    echo "ERROR: Invalid network interface ID format: $ENI_ID"
    exit 1
fi

echo "Network Interface ID: $ENI_ID"

# Get public IP
PUBLIC_IP=$(aws ec2 describe-network-interfaces --network-interface-ids "$ENI_ID"
 --query "NetworkInterfaces[0].Association.PublicIp" --output text)
if [[ "$PUBLIC_IP" == "None" || -z "$PUBLIC_IP" ]]; then
    echo "WARNING: No public IP assigned to the task"
    echo "The task may be in a private subnet or public IP assignment failed"
else
    # Validate IP format
    if [[ ! "$PUBLIC_IP" =~ ^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$ ]];
 then
        echo "ERROR: Invalid IP address format: $PUBLIC_IP"
        PUBLIC_IP=""
    else
        echo ""
        echo "========================================="
        echo "SUCCESS! APPLICATION IS RUNNING"
        echo "========================================="
        echo "Your application is available at: http://$PUBLIC_IP"
        echo "You can test it by opening this URL in your browser"
        echo ""
    fi
fi

# Display service information
echo ""
echo "========================================="
```

```
echo "SERVICE INFORMATION"
echo "=========================================="
execute_command "aws ecs describe-services --cluster '$CLUSTER_NAME' --services
 '$SERVICE_NAME'" "Get service details"

echo ""
echo "=========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY"
echo "=========================================="
echo "Resources created:"
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "  - $resource"
done

if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
    echo ""
    echo "Application URL: http://$PUBLIC_IP"
fi

echo ""
echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeNetworkInterfaces
  - DescribeSecurityGroups
  - DescribeServices
  - DescribeSubnets

- DescribeTasks
- DescribeVpcs
- GetCallerIdentity
- GetRole
- ListTaskDefinitions
- ListTasks
- RegisterTaskDefinition
- UpdateService
- Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar uma API REST com integração de proxy de função

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM para execução do Lambda.
- Criar e implantar uma função do Lambda.
- Criar uma API REST
- Configurar a integração de proxy do Lambda.
- Implantar e testar a API
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

set -euo pipefail

# Simple API Gateway Lambda Integration Script
# This script creates a REST API with Lambda proxy integration

# Generate random identifiers
FUNCTION_NAME="GetStartedLambdaProxyIntegration-$(openssl rand -hex 4)"
ROLE_NAME="GetStartedLambdaBasicExecutionRole-$(openssl rand -hex 4)"
API_NAME="LambdaProxyAPI-$(openssl rand -hex 4)"

# Get AWS account info
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
REGION=$(aws configure get region || echo "us-east-1")

# Validate inputs
if [[ -z "$ACCOUNT_ID" ]] || [[ -z "$REGION" ]]; then
    echo "Error: Failed to retrieve AWS account information" >&2
    exit 1
fi

echo "Creating Lambda function code..."

# Create Lambda function code with input validation
cat > lambda_function.py << 'EOF'
import json
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    try:
        logger.info("Received event: %s", json.dumps(event))

        greeter = 'World'

        # Safely retrieve greeter from query string parameters
        query_params = event.get('queryStringParameters') or {}
        if isinstance(query_params, dict) and 'greeter' in query_params:
            greeter_value = query_params.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
```

```
                greeter = greeter_value

        # Safely retrieve greeter from multi-value headers
        multi_headers = event.get('multiValueHeaders') or {}
        if isinstance(multi_headers, dict) and 'greeter' in multi_headers:
            greeter_list = multi_headers.get('greeter', [])
            if isinstance(greeter_list, list) and greeter_list:
                greeter = " and ".join(str(g) for g in greeter_list if g)

        # Safely retrieve greeter from headers
        headers = event.get('headers') or {}
        if isinstance(headers, dict) and 'greeter' in headers:
            greeter_value = headers.get('greeter')
            if isinstance(greeter_value, str) and greeter_value:
                greeter = greeter_value

        # Safely retrieve greeter from body
        body = event.get('body')
        if body and isinstance(body, str):
            try:
                body_dict = json.loads(body)
                if isinstance(body_dict, dict) and 'greeter' in body_dict:
                    greeter_value = body_dict.get('greeter')
                    if isinstance(greeter_value, str) and greeter_value:
                        greeter = greeter_value
            except (json.JSONDecodeError, ValueError) as e:
                logger.warning("Failed to parse body: %s", str(e))

        # Sanitize greeter to prevent injection
        greeter = greeter.replace('"', '\\"').replace("'", "\\'")

        response = {
            "statusCode": 200,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"message": f"Hello, {greeter}!"})
        }

        logger.info("Response: %s", json.dumps(response))
        return response

    except Exception as e:
        logger.error("Unexpected error: %s", str(e), exc_info=True)
```

```
        return {
            "statusCode": 500,
            "headers": {
                "Content-Type": "application/json"
            },
            "body": json.dumps({"error": "Internal server error"})
        }
EOF

# Create deployment package
zip -q function.zip lambda_function.py || {
    echo "Error: Failed to create function.zip" >&2
    exit 1
}

echo "Creating IAM role..."

# Create IAM trust policy
cat > trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role with error handling
aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document file://trust-policy.json \
    --description "Temporary role for Lambda execution" || {
    echo "Error: Failed to create IAM role" >&2
    exit 1
}

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=apigateway-lambda-integration
```

```
# Attach execution policy
aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || {
    echo "Error: Failed to attach IAM policy" >&2
    exit 1
}

# Wait for role propagation
sleep 15

echo "Creating Lambda function..."

# Create Lambda function with Python 3.11 (more recent runtime)
aws lambda create-function \
    --function-name "$FUNCTION_NAME" \
    --runtime python3.11 \
    --role "arn:aws:iam::$ACCOUNT_ID:role/$ROLE_NAME" \
    --handler lambda_function.lambda_handler \
    --zip-file fileb://function.zip \
    --timeout 30 \
    --memory-size 128 \
    --environment "Variables={LOG_LEVEL=INFO}" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration || {
    echo "Error: Failed to create Lambda function" >&2
    exit 1
}

echo "Creating API Gateway..."

# Create REST API with minimum logging
API_RESPONSE=$(aws apigateway create-rest-api \
    --name "$API_NAME" \
    --endpoint-configuration types=REGIONAL \
    --description "API for Lambda proxy integration tutorial" \
    --tags project=doc-smith,tutorial=apigateway-lambda-integration \
    --output json)

API_ID=$(echo "$API_RESPONSE" | grep -o '"id": "[^"]*"' | head -1 | cut -d'"' -
f4)

if [[ -z "$API_ID" ]]; then
```

```
    echo "Error: Failed to create API Gateway" >&2
    exit 1
fi

# Get root resource ID
ROOT_RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 'items[?path==`/`].id' --output text)

# Create helloworld resource
aws apigateway create-resource \
    --rest-api-id "$API_ID" \
    --parent-id "$ROOT_RESOURCE_ID" \
    --path-part helloworld || {
    echo "Error: Failed to create resource" >&2
    exit 1
}

# Get resource ID
RESOURCE_ID=$(aws apigateway get-resources --rest-api-id "$API_ID" --query
 "items[?pathPart=='helloworld'].id" --output text)

# Create ANY method with no authorization (intentional for tutorial)
aws apigateway put-method \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --authorization-type NONE || {
    echo "Error: Failed to create method" >&2
    exit 1
}

# Set up Lambda proxy integration
LAMBDA_URI="arn:aws:apigateway:$REGION:lambda:path/2015-03-31/functions/
arn:aws:lambda:$REGION:$ACCOUNT_ID:function:$FUNCTION_NAME/invocations"

aws apigateway put-integration \
    --rest-api-id "$API_ID" \
    --resource-id "$RESOURCE_ID" \
    --http-method ANY \
    --type AWS_PROXY \
    --integration-http-method POST \
    --uri "$LAMBDA_URI" || {
    echo "Error: Failed to create integration" >&2
    exit 1
```

```
}

# Grant API Gateway permission to invoke Lambda
STATEMENT_ID="apigateway-invoke-$(openssl rand -hex 4)"
SOURCE_ARN="arn:aws:execute-api:$REGION:$ACCOUNT_ID:$API_ID/*/*"

aws lambda add-permission \
    --function-name "$FUNCTION_NAME" \
    --statement-id "$STATEMENT_ID" \
    --action lambda:InvokeFunction \
    --principal apigateway.amazonaws.com \
    --source-arn "$SOURCE_ARN" || {
    echo "Error: Failed to add Lambda permission" >&2
    exit 1
}

# Deploy API
aws apigateway create-deployment \
    --rest-api-id "$API_ID" \
    --stage-name test \
    --description "Test deployment" || {
    echo "Error: Failed to deploy API" >&2
    exit 1
}

echo "Testing API..."

# Test the API
INVOKE_URL="https://$API_ID.execute-api.$REGION.amazonaws.com/test/helloworld"

echo "API URL: $INVOKE_URL"

# Test with query parameter (with proper URL encoding)
echo "Testing with query parameter:"
curl -s -X GET "$INVOKE_URL?greeter=John" | jq . 2>/dev/null || curl -s -X GET
 "$INVOKE_URL?greeter=John"
echo ""

# Test with header
echo "Testing with header:"
curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -H 'greeter: John' | jq . 2>/dev/null || curl -s -X GET "$INVOKE_URL" \
    -H 'content-type: application/json' \
```

```
    -H 'greeter: John'
echo ""

# Test with body
echo "Testing with POST body:"
curl -s -X POST "$INVOKE_URL" \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}' | jq . 2>/dev/null || curl -s -X POST "$INVOKE_URL"
 \
    -H 'content-type: application/json' \
    -d '{"greeter": "John"}'
echo ""

echo "Tutorial completed! API is available at: $INVOKE_URL"

# Cleanup
echo "Cleaning up resources..."

# Delete API
aws apigateway delete-rest-api --rest-api-id "$API_ID" || echo "Warning: Failed
 to delete API" >&2

# Delete Lambda function
aws lambda delete-function --function-name "$FUNCTION_NAME" || echo "Warning:
 Failed to delete Lambda function" >&2

# Detach policy and delete role
aws iam detach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" || echo "Warning: Failed to detach policy" >&2

aws iam delete-role --role-name "$ROLE_NAME" || echo "Warning: Failed to delete
 role" >&2

# Clean up local files securely
rm -f lambda_function.py function.zip trust-policy.json

echo "Cleanup completed!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddPermission
  - AttachRolePolicy

- CreateDeployment
- CreateFunction
- CreateResource
- CreateRestApi
- CreateRole
- DeleteFunction
- DeleteRestApi
- DeleteRole
- DetachRolePolicy
- GetCallerIdentity
- GetResources
- GetRestApis
- PutIntegration
- PutMethod

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar um serviço de contêiner para instâncias de máquinas virtuais

O código de exemplo abaixo mostra como:

- Criar um cluster do ECS.
- Criar e monitorar um serviço.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# ECS EC2 Launch Type Tutorial Script - UPDATED VERSION
# This script demonstrates creating an ECS cluster, launching a container
 instance,
# registering a task definition, and creating a service using the EC2 launch
 type.
# Updated to match the tutorial draft with nginx web server and service creation.
#
# - UPDATED: Changed from sleep task to nginx web server with service

set -e  # Exit on any error

# Configuration
SCRIPT_NAME="ecs-ec2-tutorial"
LOG_FILE="${SCRIPT_NAME}-$(date +%Y%m%d-%H%M%S).log"
CLUSTER_NAME="tutorial-cluster-$(openssl rand -hex 4)"
TASK_FAMILY="nginx-task-$(openssl rand -hex 4)"
SERVICE_NAME="nginx-service-$(openssl rand -hex 4)"
KEY_PAIR_NAME="ecs-tutorial-key-$(openssl rand -hex 4)"
SECURITY_GROUP_NAME="ecs-tutorial-sg-$(openssl rand -hex 4)"

# Tags
PROJECT_TAG="doc-smith"
TUTORIAL_TAG="ecs-ec2"

# Get current AWS region dynamically
AWS_REGION=$(aws configure get region || echo "us-east-1")

# Resource tracking arrays
CREATED_RESOURCES=()
CLEANUP_ORDER=()
```

```
# Logging function
log() {
    echo "[$(date '+%Y-%m-%d %H:%M:%S')] $1" | tee -a "$LOG_FILE"
}

# Error handling function
handle_error() {
    local exit_code=$?
    log "ERROR: Script failed with exit code $exit_code"
    log "ERROR: Last command: $BASH_COMMAND"

    echo ""
    echo "========================================="
    echo "ERROR OCCURRED - ATTEMPTING CLEANUP"
    echo "========================================="
    echo "Resources created before error:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done

    cleanup_resources
    exit $exit_code
}

# Set error trap
trap handle_error ERR

# FIXED: Enhanced cleanup function with proper error handling and logging
cleanup_resources() {
    log "Starting cleanup process..."
    local cleanup_errors=0

    # Delete service first (this will stop tasks automatically)
    if [[ -n "${SERVICE_ARN:-}" ]]; then
        log "Updating service to desired count 0: $SERVICE_NAME"
        if ! aws ecs update-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" --desired-count 0 2>>"$LOG_FILE"; then
            log "WARNING: Failed to update service desired count to 0"
            ((cleanup_errors++))
        else
            log "Waiting for service tasks to stop..."
            sleep 30  # Give time for tasks to stop
        fi
```

```
        log "Deleting service: $SERVICE_NAME"
        if ! aws ecs delete-service --cluster "$CLUSTER_NAME" --service
 "$SERVICE_NAME" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to delete service $SERVICE_NAME"
            ((cleanup_errors++))
        fi
    fi

    # Stop and delete any remaining tasks
    if [[ -n "${TASK_ARN:-}" ]]; then
        log "Stopping task: $TASK_ARN"
        if ! aws ecs stop-task --cluster "$CLUSTER_NAME" --task "$TASK_ARN" --
reason "Tutorial cleanup" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to stop task $TASK_ARN"
            ((cleanup_errors++))
        else
            log "Waiting for task to stop..."
            if ! aws ecs wait tasks-stopped --cluster "$CLUSTER_NAME" --tasks
 "$TASK_ARN" 2>>"$LOG_FILE"; then
                log "WARNING: Task stop wait failed for $TASK_ARN"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Deregister task definition
    if [[ -n "${TASK_DEFINITION_ARN:-}" ]]; then
        log "Deregistering task definition: $TASK_DEFINITION_ARN"
        if ! aws ecs deregister-task-definition --task-definition
 "$TASK_DEFINITION_ARN" 2>>"$LOG_FILE"; then
            log "WARNING: Failed to deregister task definition
 $TASK_DEFINITION_ARN"
            ((cleanup_errors++))
        fi
    fi

    # Terminate EC2 instance
    if [[ -n "${INSTANCE_ID:-}" ]]; then
        log "Terminating EC2 instance: $INSTANCE_ID"
        if ! aws ec2 terminate-instances --instance-ids "$INSTANCE_ID"
 2>>"$LOG_FILE"; then
            log "WARNING: Failed to terminate instance $INSTANCE_ID"
            ((cleanup_errors++))
```

```
        else
            log "Waiting for instance to terminate..."
            if ! aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"
2>>"$LOG_FILE"; then
                log "WARNING: Instance termination wait failed for $INSTANCE_ID"
                ((cleanup_errors++))
            fi
        fi
    fi

    # Delete security group with retry logic
    if [[ -n "${SECURITY_GROUP_ID:-}" ]]; then
        log "Deleting security group: $SECURITY_GROUP_ID"
        local retry_count=0
        local max_retries=3

        while [[ $retry_count -lt $max_retries ]]; do
            if aws ec2 delete-security-group --group-id "$SECURITY_GROUP_ID"
2>>"$LOG_FILE"; then
                log "Successfully deleted security group"
                break
            else
                ((retry_count++))
                if [[ $retry_count -lt $max_retries ]]; then
                    log "Retry $retry_count/$max_retries: Waiting 10 seconds
before retrying security group deletion..."
                    sleep 10
                else
                    log "ERROR: Failed to delete security group after
$max_retries attempts"
                    ((cleanup_errors++))
                fi
            fi
        done
    fi

    # Delete key pair
    if [[ -n "${KEY_PAIR_NAME:-}" ]]; then
        log "Deleting key pair: $KEY_PAIR_NAME"
        if ! aws ec2 delete-key-pair --key-name "$KEY_PAIR_NAME" 2>>"$LOG_FILE";
then
            log "WARNING: Failed to delete key pair $KEY_PAIR_NAME"
            ((cleanup_errors++))
        fi
```

```
        rm -f "${KEY_PAIR_NAME}.pem" 2>>"$LOG_FILE" || log "WARNING: Failed to
 remove local key file"
    fi

    # Delete ECS cluster
    if [[ -n "${CLUSTER_NAME:-}" ]]; then
        log "Deleting ECS cluster: $CLUSTER_NAME"
        if ! aws ecs delete-cluster --cluster "$CLUSTER_NAME" 2>>"$LOG_FILE";
 then
            log "WARNING: Failed to delete cluster $CLUSTER_NAME"
            ((cleanup_errors++))
        fi
    fi

    if [[ $cleanup_errors -eq 0 ]]; then
        log "Cleanup completed successfully"
    else
        log "Cleanup completed with $cleanup_errors warnings/errors. Check log
 file for details."
    fi
}

# Function to check prerequisites
check_prerequisites() {
    log "Checking prerequisites..."

    # Check AWS CLI
    if ! command -v aws &> /dev/null; then
        log "ERROR: AWS CLI is not installed"
        exit 1
    fi

    # Check AWS credentials
    if ! aws sts get-caller-identity &> /dev/null; then
        log "ERROR: AWS credentials not configured"
        exit 1
    fi

    # Get caller identity
    CALLER_IDENTITY=$(aws sts get-caller-identity --output text --query
 'Account')
    log "AWS Account: $CALLER_IDENTITY"
    log "AWS Region: $AWS_REGION"
```

```
    # Check for default VPC
    DEFAULT_VPC=$(aws ec2 describe-vpcs --filters "Name=is-default,Values=true"
 --query 'Vpcs[0].VpcId' --output text)
    if [[ "$DEFAULT_VPC" == "None" ]]; then
        log "ERROR: No default VPC found. Please create a VPC first."
        exit 1
    fi
    log "Using default VPC: $DEFAULT_VPC"

    # Get default subnet
    DEFAULT_SUBNET=$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=
$DEFAULT_VPC" "Name=default-for-az,Values=true" --query 'Subnets[0].SubnetId' --
output text)
    if [[ "$DEFAULT_SUBNET" == "None" ]]; then
        log "ERROR: No default subnet found"
        exit 1
    fi
    log "Using default subnet: $DEFAULT_SUBNET"

    log "Prerequisites check completed successfully"
}

# Function to create ECS cluster
create_cluster() {
    log "Creating ECS cluster: $CLUSTER_NAME"

    CLUSTER_ARN=$(aws ecs create-cluster --cluster-name "$CLUSTER_NAME" --
tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG --query
 'cluster.clusterArn' --output text)

    if [[ -z "$CLUSTER_ARN" ]]; then
        log "ERROR: Failed to create cluster"
        exit 1
    fi

    log "Created cluster: $CLUSTER_ARN"
    CREATED_RESOURCES+=("ECS Cluster: $CLUSTER_NAME")
}

# Function to create key pair
create_key_pair() {
    log "Creating EC2 key pair: $KEY_PAIR_NAME"

    # FIXED: Set secure umask before key creation
```

```
    umask 077
    aws ec2 create-key-pair --key-name "$KEY_PAIR_NAME" --query 'KeyMaterial' --
output text > "${KEY_PAIR_NAME}.pem"
    chmod 400 "${KEY_PAIR_NAME}.pem"
    umask 022  # Reset umask

    log "Created key pair: $KEY_PAIR_NAME"
    CREATED_RESOURCES+=("EC2 Key Pair: $KEY_PAIR_NAME")

    aws ec2 create-tags --resources "$KEY_PAIR_NAME" --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG 2>>"$LOG_FILE" || log "WARNING:
 Failed to tag key pair"
}

# Function to create security group
create_security_group() {
    log "Creating security group: $SECURITY_GROUP_NAME"

    SECURITY_GROUP_ID=$(aws ec2 create-security-group \
        --group-name "$SECURITY_GROUP_NAME" \
        --description "ECS tutorial security group" \
        --vpc-id "$DEFAULT_VPC" \
        --tag-specifications "ResourceType=security-
group,Tags=[{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=$TUTORIAL_TAG}]"
 \
        --query 'GroupId' --output text)

    if [[ -z "$SECURITY_GROUP_ID" ]]; then
        log "ERROR: Failed to create security group"
        exit 1
    fi

    # Add HTTP access rule for nginx web server with restricted CIDR
    # SECURITY FIX: Restrict access to specific CIDR if available, otherwise
 document the risk
    log "WARNING: Security group allows HTTP (port 80) from 0.0.0.0/0 - restrict
 this in production"
    aws ec2 authorize-security-group-ingress \
        --group-id "$SECURITY_GROUP_ID" \
        --protocol tcp \
        --port 80 \
        --cidr "0.0.0.0/0"

    log "Created security group: $SECURITY_GROUP_ID"
```

```
    log "Added HTTP access on port 80"
    CREATED_RESOURCES+=("Security Group: $SECURITY_GROUP_ID")
}

# Function to get ECS optimized AMI
get_ecs_ami() {
    log "Getting ECS-optimized AMI ID..."

    ECS_AMI_ID=$(aws ssm get-parameters \
        --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended \
        --query 'Parameters[0].Value' --output text | jq -r '.image_id')

    if [[ -z "$ECS_AMI_ID" ]]; then
        log "ERROR: Failed to get ECS-optimized AMI ID"
        exit 1
    fi

    log "ECS-optimized AMI ID: $ECS_AMI_ID"
}

# Function to create IAM role for ECS instance (if it doesn't exist)
ensure_ecs_instance_role() {
    log "Checking for ecsInstanceRole..."

    if ! aws iam get-role --role-name ecsInstanceRole &> /dev/null; then
        log "Creating ecsInstanceRole..."

        # Create trust policy
        cat > ecs-instance-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

        # SECURITY FIX: Validate JSON before using
```

```
        if ! jq empty ecs-instance-trust-policy.json 2>/dev/null; then
            log "ERROR: Invalid JSON in trust policy"
            rm -f ecs-instance-trust-policy.json
            exit 1
        fi

        # Create role
        aws iam create-role \
            --role-name ecsInstanceRole \
            --assume-role-policy-document file://ecs-instance-trust-policy.json

        aws iam tag-role --role-name ecsInstanceRole --tags Key=project,Value=
$PROJECT_TAG Key=tutorial,Value=$TUTORIAL_TAG

        # Attach managed policy
        aws iam attach-role-policy \
            --role-name ecsInstanceRole \
            --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role

        # Create instance profile
        aws iam create-instance-profile --instance-profile-name ecsInstanceRole

        # Add role to instance profile
        aws iam add-role-to-instance-profile \
            --instance-profile-name ecsInstanceRole \
            --role-name ecsInstanceRole

        # FIXED: Enhanced wait for role to be ready
        log "Waiting for IAM role to be ready..."
        aws iam wait role-exists --role-name ecsInstanceRole
        sleep 30  # Additional buffer for eventual consistency

        rm -f ecs-instance-trust-policy.json
        log "Created ecsInstanceRole"
        CREATED_RESOURCES+=("IAM Role: ecsInstanceRole")
    else
        log "ecsInstanceRole already exists"
    fi
}

# Function to launch container instance
launch_container_instance() {
    log "Launching ECS container instance..."
```

```
    # Create user data script
    cat > ecs-user-data.sh << EOF
#!/bin/bash
echo ECS_CLUSTER=$CLUSTER_NAME >> /etc/ecs/ecs.config
EOF

    # SECURITY FIX: Validate user data script before use
    if ! bash -n ecs-user-data.sh 2>/dev/null; then
        log "ERROR: Invalid user data script"
        rm -f ecs-user-data.sh
        exit 1
    fi

    INSTANCE_ID=$(aws ec2 run-instances \
        --image-id "$ECS_AMI_ID" \
        --instance-type t3.micro \
        --key-name "$KEY_PAIR_NAME" \
        --security-group-ids "$SECURITY_GROUP_ID" \
        --subnet-id "$DEFAULT_SUBNET" \
        --iam-instance-profile Name=ecsInstanceRole \
        --user-data file://ecs-user-data.sh \
        --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=ecs-
tutorial-instance},{Key=project,Value=$PROJECT_TAG},{Key=tutorial,Value=
$TUTORIAL_TAG}]" \
        --monitoring Enabled=false \
        --metadata-options HttpTokens=required,HttpPutResponseHopLimit=1 \
        --query 'Instances[0].InstanceId' --output text)

    if [[ -z "$INSTANCE_ID" ]]; then
        log "ERROR: Failed to launch EC2 instance"
        rm -f ecs-user-data.sh
        exit 1
    fi

    log "Launched EC2 instance: $INSTANCE_ID"
    log "Instance metadata options: IMDSv2 enforced with hop limit 1"
    CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

    # Wait for instance to be running
    log "Waiting for instance to be running..."
    aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

    # Wait for ECS agent to register
```

```
    log "Waiting for ECS agent to register with cluster..."
    local max_attempts=30
    local attempt=0

    while [[ $attempt -lt $max_attempts ]]; do
        CONTAINER_INSTANCES=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns' --output text)
        if [[ -n "$CONTAINER_INSTANCES" && "$CONTAINER_INSTANCES" != "None" ]];
 then
            log "Container instance registered successfully"
            break
        fi

        attempt=$((attempt + 1))
        log "Waiting for container instance registration... (attempt $attempt/
$max_attempts)"
        sleep 10
    done

    if [[ $attempt -eq $max_attempts ]]; then
        log "ERROR: Container instance failed to register within expected time"
        rm -f ecs-user-data.sh
        exit 1
    fi

    rm -f ecs-user-data.sh
}

# Function to register task definition
register_task_definition() {
    log "Creating task definition..."

    # Create nginx task definition JSON matching the tutorial
    cat > task-definition.json << 'EOF'
{
    "family": "TASK_FAMILY_PLACEHOLDER",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
```

```
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/nginx-task",
                    "awslogs-region": "REGION_PLACEHOLDER",
                    "awslogs-stream-prefix": "ecs"
                }
            }
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge",
    "tags": [
        {
            "key": "project",
            "value": "PROJECT_TAG_PLACEHOLDER"
        },
        {
            "key": "tutorial",
            "value": "TUTORIAL_TAG_PLACEHOLDER"
        }
    ]
}
EOF

    # Replace placeholders securely
    sed -i "s|TASK_FAMILY_PLACEHOLDER|$TASK_FAMILY|g" task-definition.json
    sed -i "s|REGION_PLACEHOLDER|$AWS_REGION|g" task-definition.json
    sed -i "s|PROJECT_TAG_PLACEHOLDER|$PROJECT_TAG|g" task-definition.json
    sed -i "s|TUTORIAL_TAG_PLACEHOLDER|$TUTORIAL_TAG|g" task-definition.json

    # FIXED: Validate JSON before registration
    if ! jq empty task-definition.json 2>/dev/null; then
        log "ERROR: Invalid JSON in task definition"
        rm -f task-definition.json
        exit 1
    fi
```

```
    TASK_DEFINITION_ARN=$(aws ecs register-task-definition \
        --cli-input-json file://task-definition.json \
        --query 'taskDefinition.taskDefinitionArn' --output text)

    if [[ -z "$TASK_DEFINITION_ARN" ]]; then
        log "ERROR: Failed to register task definition"
        rm -f task-definition.json
        exit 1
    fi

    log "Registered task definition: $TASK_DEFINITION_ARN"
    log "Task definition includes CloudWatch Logs configuration for monitoring"
    CREATED_RESOURCES+=("Task Definition: $TASK_DEFINITION_ARN")

    rm -f task-definition.json
}

# Function to create service
create_service() {
    log "Creating ECS service..."

    SERVICE_ARN=$(aws ecs create-service \
        --cluster "$CLUSTER_NAME" \
        --service-name "$SERVICE_NAME" \
        --task-definition "$TASK_FAMILY" \
        --desired-count 1 \
        --tags key=project,value=$PROJECT_TAG key=tutorial,value=$TUTORIAL_TAG \
        --query 'service.serviceArn' --output text)

    if [[ -z "$SERVICE_ARN" ]]; then
        log "ERROR: Failed to create service"
        exit 1
    fi

    log "Created service: $SERVICE_ARN"
    CREATED_RESOURCES+=("ECS Service: $SERVICE_NAME")

    # Wait for service to be stable
    log "Waiting for service to be stable..."
    aws ecs wait services-stable --cluster "$CLUSTER_NAME" --services
 "$SERVICE_NAME"

    log "Service is now stable and running"
```

```
    # Get the task ARN for monitoring
    TASK_ARN=$(aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name
 "$SERVICE_NAME" --query 'taskArns[0]' --output text)
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Service task: $TASK_ARN"
        CREATED_RESOURCES+=("ECS Task: $TASK_ARN")
    fi
}

# Function to demonstrate monitoring and testing
demonstrate_monitoring() {
    log "Demonstrating monitoring capabilities..."

    # List services
    log "Listing services in cluster:"
    aws ecs list-services --cluster "$CLUSTER_NAME" --output table

    # Describe service
    log "Service details:"
    aws ecs describe-services --cluster "$CLUSTER_NAME" --
services "$SERVICE_NAME" --output table --query 'services[0].
{ServiceName:serviceName,Status:status,RunningCount:runningCount,DesiredCount:desiredCoun

    # List tasks
    log "Listing tasks in service:"
    aws ecs list-tasks --cluster "$CLUSTER_NAME" --service-name "$SERVICE_NAME"
 --output table

    # Describe task
    if [[ -n "$TASK_ARN" && "$TASK_ARN" != "None" ]]; then
        log "Task details:"
        aws ecs describe-tasks --cluster "$CLUSTER_NAME"
 --tasks "$TASK_ARN" --output table --query 'tasks[0].
{TaskArn:taskArn,LastStatus:lastStatus,DesiredStatus:desiredStatus,CreatedAt:createdAt}'
    fi

    # List container instances
    log "Container instances in cluster:"
    aws ecs list-container-instances --cluster "$CLUSTER_NAME" --output table

    # Describe container instance
    CONTAINER_INSTANCE_ARN=$(aws ecs list-container-instances --cluster
 "$CLUSTER_NAME" --query 'containerInstanceArns[0]' --output text)
```

```
    if [[ -n "$CONTAINER_INSTANCE_ARN" && "$CONTAINER_INSTANCE_ARN" != "None" ]];
 then
        log "Container instance details:"
        aws ecs describe-container-instances --cluster "$CLUSTER_NAME"
 --container-instances "$CONTAINER_INSTANCE_ARN" --output table --query
 'containerInstances[0].
{Arn:containerInstanceArn,Status:status,RunningTasks:runningTasksCount,PendingTasks:pendi
    fi

    # Test the nginx web server
    log "Testing nginx web server..."
    PUBLIC_IP=$(aws ec2 describe-instances --instance-ids "$INSTANCE_ID" --query
 'Reservations[0].Instances[0].PublicIpAddress' --output text)

    if [[ -n "$PUBLIC_IP" && "$PUBLIC_IP" != "None" ]]; then
        log "Container instance public IP: $PUBLIC_IP"
        log "Testing HTTP connection to nginx..."

        # Wait a moment for nginx to be fully ready
        sleep 10

        if curl -s --connect-timeout 10 "http://$PUBLIC_IP" | grep -q "Welcome to
 nginx"; then
            log "SUCCESS: Nginx web server is responding correctly"
            echo ""
            echo "============================================="
            echo "WEB SERVER TEST SUCCESSFUL"
            echo "============================================="
            echo "You can access your nginx web server at: http://$PUBLIC_IP"
            echo "The nginx welcome page should be visible in your browser."
        else
            log "WARNING: Nginx web server may not be fully ready yet. Try
 accessing http://$PUBLIC_IP in a few minutes."
        fi
    else
        log "WARNING: Could not retrieve public IP address"
    fi
}

# Main execution
main() {
    log "Starting ECS EC2 Launch Type Tutorial (UPDATED VERSION)"
    log "Log file: $LOG_FILE"
```

```
    log "Security improvements: IMDSv2 enforced, JSON validation, input
 sanitization, CloudWatch Logs configured"

    check_prerequisites
    create_cluster
    create_key_pair
    create_security_group
    get_ecs_ami
    ensure_ecs_instance_role
    launch_container_instance
    register_task_definition
    create_service
    demonstrate_monitoring

    log "Tutorial completed successfully!"

    echo ""
    echo "========================================="
    echo "TUTORIAL COMPLETED SUCCESSFULLY"
    echo "========================================="
    echo "Resources created:"
    for resource in "${CREATED_RESOURCES[@]}"; do
        echo "  - $resource"
    done
    echo ""
    echo "The nginx service will continue running and maintain the desired task
 count."
    echo "You can monitor the service status using:"
    echo "  aws ecs describe-services --cluster $CLUSTER_NAME --services
 $SERVICE_NAME"
    echo ""
    if [[ -n "${PUBLIC_IP:-}" ]]; then
        echo "Access your web server at: http://$PUBLIC_IP"
        echo ""
    fi
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    log "Auto-confirming cleanup - proceeding with resource cleanup"
    cleanup_resources
    log "All resources have been cleaned up"
}

# Run main function
```

```
main "$@"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreateRole
  - CreateSecurityGroup
  - CreateService
  - DeleteCluster
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DeleteService
  - DeregisterTaskDefinition
  - DescribeContainerInstances
  - DescribeInstances
  - DescribeServices
  - DescribeSubnets
  - DescribeTasks
  - DescribeVpcs
  - GetCallerIdentity
  - GetParameters
  - GetRole
  - ListContainerInstances
  - ListServices
  - ListTasks
  - RegisterTaskDefinition
  - RunInstances

- StopTask
- TerminateInstances
- UpdateService
- Aguardar (Amazon EC2)
- Aguardar (Amazon ECS)
- Aguardar (IAM)

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar um espaço de trabalho de monitoramento gerenciado

O código de exemplo abaixo mostra como:

- Criar um perfil do IAM para seu espaço de trabalho.
- Criar um espaço de trabalho do Grafana.
- Configurar autenticação
- Definir configurações opcionais.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon Managed Grafana Workspace Creation Script
# This script creates an Amazon Managed Grafana workspace and configures it
```

```
# Set up logging
LOG_FILE="grafana-workspace-creation.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon Managed Grafana workspace creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error\|exception\|fail" > /dev/null; then
        echo "ERROR: Command '$cmd' failed with output:"
        echo "$output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..."

    if [ -n "$WORKSPACE_ID" ]; then
        echo "Deleting workspace $WORKSPACE_ID..."
        aws grafana delete-workspace --workspace-id "$WORKSPACE_ID"
    fi

    if [ -n "$ROLE_NAME" ]; then
        echo "Detaching policies from role $ROLE_NAME..."
        if [ -n "$POLICY_ARN" ]; then
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$POLICY_ARN"
        fi

        echo "Deleting role $ROLE_NAME..."
        aws iam delete-role --role-name "$ROLE_NAME"
    fi

    if [ -n "$POLICY_ARN" ]; then
        echo "Deleting policy..."
        aws iam delete-policy --policy-arn "$POLICY_ARN"
    fi
```

```
    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed. See $LOG_FILE for details."
}

# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
WORKSPACE_NAME="GrafanaWorkspace-${RANDOM_ID}"
ROLE_NAME="GrafanaWorkspaceRole-${RANDOM_ID}"

echo "Using workspace name: $WORKSPACE_NAME"
echo "Using role name: $ROLE_NAME"

# Step 1: Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "AWS Account ID: $ACCOUNT_ID"

# Step 2: Create IAM role for Grafana workspace
echo "Creating IAM role for Grafana workspace..."

# Create trust policy document
cat > trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "grafana.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role
ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$ROLE_NAME" \
  --assume-role-policy-document file://trust-policy.json \
```

```
  --description "Role for Amazon Managed Grafana workspace")

check_error "$ROLE_OUTPUT" "create-role"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-grafana-gs
echo "IAM role created successfully"

# Extract role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "Role ARN: $ROLE_ARN"

# Attach policies to the role
echo "Attaching policies to the role..."

# CloudWatch policy
cat > cloudwatch-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    }
  ]
}
EOF

POLICY_OUTPUT=$(aws iam create-policy \
  --policy-name "GrafanaCloudWatchPolicy-${RANDOM_ID}" \
  --policy-document file://cloudwatch-policy.json)

check_error "$POLICY_OUTPUT" "create-policy"

POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)
echo "CloudWatch policy ARN: $POLICY_ARN"
```

```
ATTACH_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$ROLE_NAME" \
  --policy-arn "$POLICY_ARN")

check_error "$ATTACH_OUTPUT" "attach-role-policy"
echo "CloudWatch policy attached to role"

# Step 3: Create the Grafana workspace
echo "Creating Amazon Managed Grafana workspace..."
WORKSPACE_OUTPUT=$(aws grafana create-workspace \
  --workspace-name "$WORKSPACE_NAME" \
  --authentication-providers "SAML" \
  --permission-type "CUSTOMER_MANAGED" \
  --account-access-type "CURRENT_ACCOUNT" \
  --workspace-role-arn "$ROLE_ARN" \
  --workspace-data-sources "CLOUDWATCH" "PROMETHEUS" "XRAY" \
  --grafana-version "10.4" \
  --tags Environment=Development,project=doc-smith,tutorial=amazon-managed-
grafana-gs)

check_error "$WORKSPACE_OUTPUT" "create-workspace"

echo "Workspace creation initiated:"
echo "$WORKSPACE_OUTPUT"

# Extract workspace ID
WORKSPACE_ID=$(echo "$WORKSPACE_OUTPUT" | grep -o '"id": "[^"]*' | cut -d'"' -f4)

if [ -z "$WORKSPACE_ID" ]; then
    echo "ERROR: Failed to extract workspace ID from output"
    exit 1
fi

echo "Workspace ID: $WORKSPACE_ID"

# Step 4: Wait for workspace to become active
echo "Waiting for workspace to become active. This may take several minutes..."
ACTIVE=false
MAX_ATTEMPTS=30
ATTEMPT=0

while [ $ACTIVE = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
    ATTEMPT=$((ATTEMPT+1))
    echo "Checking workspace status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."
```

```
    DESCRIBE_OUTPUT=$(aws grafana describe-workspace --workspace-id
 "$WORKSPACE_ID")
    check_error "$DESCRIBE_OUTPUT" "describe-workspace"

    STATUS=$(echo "$DESCRIBE_OUTPUT" | grep -o '"status": "[^"]*' | cut -d'"' -
f4)
    echo "Current status: $STATUS"

    if [ "$STATUS" = "ACTIVE" ]; then
        ACTIVE=true
        echo "Workspace is now ACTIVE"
    elif [ "$STATUS" = "FAILED" ]; then
        echo "ERROR: Workspace creation failed"
        cleanup_on_error
        exit 1
    else
        echo "Workspace is still being created. Waiting 30 seconds..."
        sleep 30
    fi
done

if [ $ACTIVE = false ]; then
    echo "ERROR: Workspace did not become active within the expected time"
    cleanup_on_error
    exit 1
fi

# Extract workspace endpoint URL
WORKSPACE_URL=$(echo "$DESCRIBE_OUTPUT" | grep -o '"endpoint": "[^"]*' | cut -
d'"' -f4)
echo "Workspace URL: https://$WORKSPACE_URL"

# Step 5: Display workspace information
echo ""
echo "========================================="
echo "WORKSPACE INFORMATION"
echo "========================================="
echo "Workspace ID: $WORKSPACE_ID"
echo "Workspace URL: https://$WORKSPACE_URL"
echo "Workspace Name: $WORKSPACE_NAME"
echo "IAM Role: $ROLE_NAME"
echo ""
```

```
echo "Note: Since SAML authentication is used, you need to configure SAML
 settings"
echo "using the AWS Management Console or the update-workspace-authentication
 command."
echo "========================================="

# Step 6: Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Resources created:"
echo "- Amazon Managed Grafana workspace: $WORKSPACE_ID"
echo "- IAM Role: $ROLE_NAME"
echo "- IAM Policy: GrafanaCloudWatchPolicy-${RANDOM_ID}"
echo ""
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy] ]]; then
    echo "Cleaning up resources..."

    echo "Deleting workspace $WORKSPACE_ID..."
    DELETE_OUTPUT=$(aws grafana delete-workspace --workspace-id "$WORKSPACE_ID")
    check_error "$DELETE_OUTPUT" "delete-workspace"

    echo "Waiting for workspace to be deleted..."
    DELETED=false
    ATTEMPT=0

    while [ $DELETED = false ] && [ $ATTEMPT -lt $MAX_ATTEMPTS ]; do
        ATTEMPT=$((ATTEMPT+1))
        echo "Checking deletion status (attempt $ATTEMPT of $MAX_ATTEMPTS)..."

        if aws grafana describe-workspace --workspace-id "$WORKSPACE_ID" 2>&1 |
 grep -i "not found\|does not exist" > /dev/null; then
            DELETED=true
            echo "Workspace has been deleted"
        else
            echo "Workspace is still being deleted. Waiting 30 seconds..."
            sleep 30
        fi
    done
```

```
    if [ $DELETED = false ]; then
        echo "WARNING: Workspace deletion is taking longer than expected. It may
 still be in progress."
    fi

    # Detach policy from role
    echo "Detaching policy from role..."
    aws iam detach-role-policy \
      --role-name "$ROLE_NAME" \
      --policy-arn "$POLICY_ARN"

    # Delete policy
    echo "Deleting IAM policy..."
    aws iam delete-policy \
      --policy-arn "$POLICY_ARN"

    # Delete role
    echo "Deleting IAM role..."
    aws iam delete-role \
      --role-name "$ROLE_NAME"

    # Clean up JSON files
    rm -f trust-policy.json cloudwatch-policy.json

    echo "Cleanup completed"
else
    echo "Skipping cleanup. Resources will remain in your AWS account."
fi

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreatePolicy
  - CreateRole
  - CreateWorkspace
  - DeletePolicy
  - DeleteRole
  - DeleteWorkspace

- DescribeWorkspace
- DetachRolePolicy
- GetCallerIdentity

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como criar um painel de monitoramento com o nome da função como uma variável

O código de exemplo abaixo mostra como:

- Criar um painel do CloudWatch
- Adicionar widgets de métricas do Lambda com uma variável de nome de função.
- Verificar o painel.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Script to create a CloudWatch dashboard with Lambda function name as a variable
# This script creates a CloudWatch dashboard that allows you to switch between
 different Lambda functions

# Set up logging with secure permissions
LOG_FILE="${HOME}/.cloudwatch-dashboard-script.log"
touch "$LOG_FILE" && chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1
```

```
echo "$(date): Starting CloudWatch dashboard creation script"

# Security: Set strict error handling
set -uo pipefail
trap 'handle_error "Script failed at line $LINENO"' ERR

# Function to handle errors
handle_error() {
    local error_msg="${1:-Unknown error}"
    echo "ERROR: $error_msg" >&2
    echo "Resources created:"
    echo "- CloudWatch Dashboard: LambdaMetricsDashboard"
    echo ""
    echo "=========================================="
    echo "CLEANUP CONFIRMATION"
    echo "=========================================="
    echo "An error occurred. Proceeding with automatic cleanup..."

    echo "Cleaning up resources..."
    aws cloudwatch delete-dashboards --dashboard-names LambdaMetricsDashboard 2>/
dev/null || true

    # Clean up temporary files
    if [ -n "${TEMP_DIR:-}" ] && [ -d "$TEMP_DIR" ]; then
        rm -rf "$TEMP_DIR"
    fi
    rm -f dashboard-body.json

    echo "Cleanup complete."
    exit 1
}

# Security: Validate AWS CLI is installed
if ! command -v aws &> /dev/null; then
    handle_error "AWS CLI is not installed. Please install it and try again."
fi

# Check if AWS CLI is installed and configured
echo "Checking AWS CLI configuration..."
if ! aws sts get-caller-identity > /dev/null 2>&1; then
    handle_error "AWS CLI is not properly configured. Please configure it with
 'aws configure' and try again."
fi
```

```
# Get the current region securely
REGION=$(aws configure get region 2>/dev/null || echo "")
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $REGION"
fi
echo "Using region: $REGION"

# Validate region format
if ! [[ "$REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]{1}$ ]]; then
    handle_error "Invalid AWS region format: $REGION"
fi

# Check if there are any Lambda functions in the account
echo "Checking for Lambda functions..."
LAMBDA_FUNCTIONS=$(aws lambda list-functions --region "$REGION" --query
 "Functions[*].FunctionName" --output text 2>/dev/null || echo "")

if [ -z "$LAMBDA_FUNCTIONS" ]; then
    echo "No Lambda functions found in your account. Creating a simple test
 function..."

    # Create a temporary directory for Lambda function code with secure
 permissions
    TEMP_DIR=$(mktemp -d)
    chmod 700 "$TEMP_DIR"
    trap 'rm -rf "$TEMP_DIR"' EXIT

    # Create a simple Lambda function
    cat > "$TEMP_DIR/index.js" << 'EOF'
exports.handler = async (event) => {
    console.log('Event:', JSON.stringify(event, null, 2));
    return {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
};
EOF

    # Zip the function code
    if ! cd "$TEMP_DIR"; then
        handle_error "Failed to change to temporary directory"
    fi
```

```
    if ! zip -q function.zip index.js; then
        handle_error "Failed to create zip file"
    fi

    # Create a role for the Lambda function with restricted trust policy
    ROLE_NAME="LambdaDashboardTestRole-$(date +%s)"
    TRUST_POLICY='{"Version":"2012-10-17",       "Statement":
[{"Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}'

    if ! ROLE_ARN=$(aws iam create-role \
        --role-name "$ROLE_NAME" \
        --assume-role-policy-document "$TRUST_POLICY" \
        --tags Key=project,Value=doc-smith Key=tutorial,Value=cloudwatch-
dynamicdash \
        --query "Role.Arn" \
        --output text 2>/dev/null); then
        handle_error "Failed to create IAM role for Lambda function"
    fi

    echo "Waiting for role to be available..."
    sleep 10

    # Attach basic Lambda execution policy
    if ! aws iam attach-role-policy \
        --role-name "$ROLE_NAME" \
        --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"; then
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to attach policy to IAM role"
    fi

    # Create the Lambda function
    FUNCTION_NAME="DashboardTestFunction-$(date +%s)"
    if ! aws lambda create-function \
        --function-name "$FUNCTION_NAME" \
        --runtime nodejs18.x \
        --role "$ROLE_ARN" \
        --handler index.handler \
        --zip-file fileb://function.zip \
        --tags project=doc-smith,tutorial=cloudwatch-dynamicdash \
        --region "$REGION" > /dev/null 2>&1; then
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
```

```
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
        handle_error "Failed to create Lambda function"
    fi

    # Invoke the function to generate some metrics
    echo "Invoking Lambda function to generate metrics..."
    for i in {1..5}; do
        aws lambda invoke --function-name "$FUNCTION_NAME" --payload '{}' /dev/
null --region "$REGION" > /dev/null 2>&1 || true
        sleep 1
    done

    # Go back to original directory
    cd - > /dev/null

    # Set the function name for the dashboard
    DEFAULT_FUNCTION="$FUNCTION_NAME"
else
    # Use the first Lambda function as default
    DEFAULT_FUNCTION=$(echo "$LAMBDA_FUNCTIONS" | awk '{print $1}')
    echo "Found Lambda functions. Using $DEFAULT_FUNCTION as default."
    FUNCTION_NAME=""
    ROLE_NAME=""
fi

# Create a dashboard with Lambda metrics and a function name variable
echo "Creating CloudWatch dashboard with Lambda function name variable..."

# Create a JSON file for the dashboard body with secure permissions
DASHBOARD_JSON="dashboard-body-$$.json"
touch "$DASHBOARD_JSON" && chmod 600 "$DASHBOARD_JSON"

# Escape special characters in region and function name for JSON
REGION_ESCAPED=$(printf '%s\n' "$REGION" | sed 's:[\/&]:\\&:g')
FUNCTION_ESCAPED=$(printf '%s\n' "$DEFAULT_FUNCTION" | sed 's:[\/&]:\\&:g')

cat > "$DASHBOARD_JSON" << EOF
{
  "widgets": [
    {
      "type": "metric",
      "x": 0,
```

```
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Invocations", "FunctionName", "\${FunctionName}" ],
          [ ".", "Errors", ".", "." ],
          [ ".", "Throttles", ".", "." ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Lambda Function Metrics for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 0,
      "y": 6,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "Duration", "FunctionName", "\${FunctionName}",
 { "stat": "Average" } ]
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Duration for \${FunctionName}",
        "period": 300
      }
    },
    {
      "type": "metric",
      "x": 12,
      "y": 0,
      "width": 12,
      "height": 6,
      "properties": {
        "metrics": [
          [ "AWS/Lambda", "ConcurrentExecutions", "FunctionName", "\
${FunctionName}" ]
```

```
        ],
        "view": "timeSeries",
        "stacked": false,
        "region": "$REGION_ESCAPED",
        "title": "Concurrent Executions for \${FunctionName}",
        "period": 300
      }
    }
  ],
  "periodOverride": "auto",
  "variables": [
    {
      "type": "property",
      "id": "FunctionName",
      "property": "FunctionName",
      "label": "Lambda Function",
      "inputType": "select",
      "values": [
        {
          "value": "$FUNCTION_ESCAPED",
          "label": "$FUNCTION_ESCAPED"
        }
      ]
    }
  ]
}
EOF

# Validate JSON before sending
if ! jq empty "$DASHBOARD_JSON" 2>/dev/null; then
    handle_error "Invalid JSON generated for dashboard"
fi

# Create the dashboard using the JSON file
DASHBOARD_NAME="LambdaMetricsDashboard-$(date +%s)"
if ! DASHBOARD_RESULT=$(aws cloudwatch put-dashboard \
    --dashboard-name "$DASHBOARD_NAME" \
    --dashboard-body file://"$DASHBOARD_JSON" \
    --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
```

```
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to create CloudWatch dashboard."
fi

# Display any validation messages but continue
if echo "$DASHBOARD_RESULT" | grep -q "DashboardValidationMessages"; then
    echo "Dashboard created with validation messages:"
    echo "$DASHBOARD_RESULT"
    echo "These validation messages are warnings and the dashboard should still
 function."
else
    echo "Dashboard created successfully!"
fi

# Verify the dashboard was created
echo "Verifying dashboard creation..."
if ! DASHBOARD_INFO=$(aws cloudwatch get-dashboard --dashboard-name
 "$DASHBOARD_NAME" --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to verify dashboard creation."
fi

echo "Dashboard verification successful!"
echo "Dashboard details:"
echo "$DASHBOARD_INFO"

# List all dashboards to confirm
echo "Listing all dashboards:"
if ! DASHBOARDS=$(aws cloudwatch list-dashboards --region "$REGION" 2>&1); then
    # If we created resources, clean them up
    if [ -n "${FUNCTION_NAME:-}" ]; then
```

```
        aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null || true
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null || true
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi
    handle_error "Failed to list dashboards."
fi
echo "$DASHBOARDS"

# Show instructions for accessing the dashboard
echo ""
echo "Dashboard created successfully! To access it:"
echo "1. Open the CloudWatch console at https://console.aws.amazon.com/
cloudwatch/"
echo "2. In the navigation pane, choose Dashboards"
echo "3. Select $DASHBOARD_NAME"
echo "4. You should see a dropdown menu labeled 'Lambda Function' at the top of
 the dashboard"
echo "5. Use this dropdown to select different Lambda functions and see their
 metrics"
echo ""

# Create a list of resources for cleanup
RESOURCES=("- CloudWatch Dashboard: $DASHBOARD_NAME")
if [ -n "${FUNCTION_NAME:-}" ]; then
    RESOURCES+=("- Lambda Function: $FUNCTION_NAME")
    RESOURCES+=("- IAM Role: $ROLE_NAME")
fi

# Prompt for cleanup with automatic yes
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Resources created:"
for resource in "${RESOURCES[@]}"; do
    echo "$resource"
done
echo ""
echo "Proceeding with automatic cleanup..."

CLEANUP_CHOICE="y"
```

```
if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."

    # Delete the dashboard
    if aws cloudwatch delete-dashboards --dashboard-names "$DASHBOARD_NAME" --
region "$REGION" 2>/dev/null; then
        echo "Dashboard deleted successfully."
    else
        echo "WARNING: Failed to delete dashboard. You may need to delete it
 manually."
    fi

    # If we created a Lambda function, delete it and its role
    if [ -n "${FUNCTION_NAME:-}" ]; then
        echo "Deleting Lambda function..."
        if aws lambda delete-function --function-name "$FUNCTION_NAME" --region
 "$REGION" 2>/dev/null; then
            echo "Lambda function deleted successfully."
        else
            echo "WARNING: Failed to delete Lambda function. You may need to
 delete it manually."
        fi

        echo "Detaching role policy..."
        if aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole" 2>/dev/null; then
            echo "Role policy detached successfully."
        else
            echo "WARNING: Failed to detach role policy. You may need to detach
 it manually."
        fi

        echo "Deleting IAM role..."
        if aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null; then
            echo "IAM role deleted successfully."
        else
            echo "WARNING: Failed to delete IAM role. You may need to delete it
 manually."
        fi
    fi
```

```
    # Clean up the JSON file
    rm -f "$DASHBOARD_JSON"

    echo "Cleanup complete."
fi

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateFunction
  - CreateRole
  - DeleteDashboards
  - DeleteFunction
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetDashboard
  - Invocar
  - ListDashboards
  - ListFunctions
  - PutDashboard

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK

O exemplo de código a seguir mostra como obter um token de sessão que exige um token de MFA.

> **Warning**
>
> Para evitar riscos de segurança, não use usuários do IAM para autenticação ao desenvolver software com propósito específico ou trabalhar com dados reais. Em vez disso, use federação com um provedor de identidade, como Centro de Identidade do AWS IAM.

- Criar um perfil do IAM que conceda permissão para listar os buckets do Amazon S3.
- Criar um usuário do IAM que tenha permissão para assumir o perfil somente quando as credenciais de MFA forem fornecidas.
- Registrar um dispositivo MFA para o usuário.
- Forneça credenciais de MFA para obter um token de sessão e use credenciais temporárias para listar buckets do S3.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

Crie um usuário do IAM, registre um dispositivo de MFA e crie um perfil que conceda permissão para deixar o usuário listar os buckets do S3 somente quando credenciais de MFA forem usadas.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.
```

```
    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)
```

```
    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version":"2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
 True}},
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Obtenha credenciais de sessão temporárias passando um token de MFA e use-as para listar os buckets do S3 para a conta.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.
```

```
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Destrua os recursos criados para a demonstração.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
```

```
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Execute esse cenário usando a funções definidas anteriormente.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Para obter detalhes da API, consulte GetSessionToken na Referência da API do AWS SDK for Python (Boto3).

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos do gerenciamento de configurações

O código de exemplo abaixo mostra como:

- Criar um bucket do Amazon S3.
- Criar um tópico do Amazon SNS.
- Criar um perfil do IAM para o Config.
- Configurar o gravador de configuração do Config.
- Configurar o canal de entrega do Config.
- Iniciar o gravador de configuração.
- Verificar a configuração do Config

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS Config Setup Script (v2)
# This script sets up AWS Config with the AWS CLI

# Error handling
set -e
LOGFILE="aws-config-setup-v2.log"
touch $LOGFILE
```

```
exec > >(tee -a $LOGFILE)
exec 2>&1

# Function to handle errors
handle_error() {
    echo "ERROR: An error occurred at line $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Set trap for error handling
trap 'handle_error $LINENO' ERR

# Function to generate random identifier
generate_random_id() {
    echo $(openssl rand -hex 6)
}

# Function to check if command was successful
check_command() {
    if echo "$1" | grep -i "error" > /dev/null; then
        echo "ERROR: $1"
        return 1
    fi
    return 0
}

# Function to clean up resources
cleanup_resources() {
    if [ -n "$CONFIG_RECORDER_NAME" ]; then
        echo "Stopping configuration recorder..."
        aws configservice stop-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    # Check if we created a new delivery channel before trying to delete it
    if [ -n "$DELIVERY_CHANNEL_NAME" ] && [ "$CREATED_NEW_DELIVERY_CHANNEL" =
 "true" ]; then
        echo "Deleting delivery channel..."
        aws configservice delete-delivery-channel --delivery-channel-name
 "$DELIVERY_CHANNEL_NAME" 2>/dev/null || true
    fi
```

```
    if [ -n "$CONFIG_RECORDER_NAME" ] && [ "$CREATED_NEW_CONFIG_RECORDER" =
 "true" ]; then
        echo "Deleting configuration recorder..."
        aws configservice delete-configuration-recorder --configuration-recorder-
name "$CONFIG_RECORDER_NAME" 2>/dev/null || true
    fi

    if [ -n "$ROLE_NAME" ]; then
        if [ -n "$POLICY_NAME" ]; then
            echo "Detaching custom policy from role..."
            aws iam delete-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" 2>/dev/null || true
        fi

        if [ -n "$MANAGED_POLICY_ARN" ]; then
            echo "Detaching managed policy from role..."
            aws iam detach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN" 2>/dev/null || true
        fi

        echo "Deleting IAM role..."
        aws iam delete-role --role-name "$ROLE_NAME" 2>/dev/null || true
    fi

    if [ -n "$SNS_TOPIC_ARN" ]; then
        echo "Deleting SNS topic..."
        aws sns delete-topic --topic-arn "$SNS_TOPIC_ARN" 2>/dev/null || true
    fi

    if [ -n "$S3_BUCKET_NAME" ]; then
        echo "Emptying S3 bucket..."
        aws s3 rm "s3://$S3_BUCKET_NAME" --recursive 2>/dev/null || true

        echo "Deleting S3 bucket..."
        if [ "$BUCKET_IS_SHARED" = "false" ]; then
            aws s3api delete-bucket --bucket "$S3_BUCKET_NAME" 2>/dev/null ||
 true
        fi
    fi
}

# Function to display created resources
display_resources() {
    echo ""
```

```
    echo "========================================="
    echo "CREATED RESOURCES"
    echo "========================================="
    echo "S3 Bucket: $S3_BUCKET_NAME"
    echo "SNS Topic ARN: $SNS_TOPIC_ARN"
    echo "IAM Role: $ROLE_NAME"
    if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (newly created)"
    else
        echo "Configuration Recorder: $CONFIG_RECORDER_NAME (existing)"
    fi
    if [ "$CREATED_NEW_DELIVERY_CHANNEL" = "true" ]; then
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (newly created)"
    else
        echo "Delivery Channel: $DELIVERY_CHANNEL_NAME (existing)"
    fi
    echo "========================================="
}

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
if [ -z "$ACCOUNT_ID" ]; then
    echo "ERROR: Failed to get AWS account ID"
    exit 1
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Generate random identifier for resources
RANDOM_ID=$(generate_random_id)
echo "Generated random identifier: $RANDOM_ID"

# Step 1: Create an S3 bucket
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
```

```
    S3_BUCKET_NAME="configservice-${RANDOM_ID}"
    echo "Creating S3 bucket: $S3_BUCKET_NAME"
fi

# Get the current region
AWS_REGION=$(aws configure get region)
if [ -z "$AWS_REGION" ]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
fi
echo "Using AWS Region: $AWS_REGION"

# Create bucket with appropriate command based on region
if [ "$BUCKET_IS_SHARED" = "false" ]; then
    if [ "$AWS_REGION" = "us-east-1" ]; then
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME")
    else
        BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" --
create-bucket-configuration LocationConstraint="$AWS_REGION")
    fi
    check_command "$BUCKET_RESULT"
    echo "S3 bucket created: $S3_BUCKET_NAME"

    aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-config-gs}]'
    echo "Tags applied to S3 bucket"
else
    echo "Using shared bucket: $S3_BUCKET_NAME (skipping creation)"
fi

# Block public access for the bucket
aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
echo "Public access blocked for bucket"

# Step 2: Create an SNS topic
TOPIC_NAME="config-topic-${RANDOM_ID}"
echo "Creating SNS topic: $TOPIC_NAME"
SNS_RESULT=$(aws sns create-topic --name "$TOPIC_NAME" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs)
check_command "$SNS_RESULT"
SNS_TOPIC_ARN=$(echo "$SNS_RESULT" | grep -o 'arn:aws:sns:[^"]*')
echo "SNS topic created: $SNS_TOPIC_ARN"
```

```
# Step 3: Create an IAM role for AWS Config
ROLE_NAME="config-role-${RANDOM_ID}"
POLICY_NAME="config-delivery-permissions"
MANAGED_POLICY_ARN="arn:aws:iam::aws:policy/service-role/AWS_ConfigRole"

echo "Creating trust policy document..."
cat > config-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "config.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

echo "Creating IAM role: $ROLE_NAME"
ROLE_RESULT=$(aws iam create-role --role-name "$ROLE_NAME" --assume-role-policy-
document file://config-trust-policy.json)
check_command "$ROLE_RESULT"
ROLE_ARN=$(echo "$ROLE_RESULT" | grep -o 'arn:aws:iam::[^"]*' | head -1)
echo "IAM role created: $ROLE_ARN"

aws iam tag-role --role-name "$ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-config-gs
echo "Tags applied to IAM role"

echo "Attaching AWS managed policy to role..."
ATTACH_RESULT=$(aws iam attach-role-policy --role-name "$ROLE_NAME" --policy-arn
 "$MANAGED_POLICY_ARN")
check_command "$ATTACH_RESULT"
echo "AWS managed policy attached"

echo "Creating custom policy document for S3 and SNS access..."
cat > config-delivery-permissions.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}/AWSLogs/${ACCOUNT_ID}/*",
      "Condition": {
        "StringLike": {
          "s3:x-amz-acl": "bucket-owner-full-control"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl"
      ],
      "Resource": "arn:aws:s3:::${S3_BUCKET_NAME}"
    },
    {
      "Effect": "Allow",
      "Action": [
        "sns:Publish"
      ],
      "Resource": "${SNS_TOPIC_ARN}"
    }
  ]
}
EOF

echo "Attaching custom policy to role..."
POLICY_RESULT=$(aws iam put-role-policy --role-name "$ROLE_NAME" --policy-name
 "$POLICY_NAME" --policy-document file://config-delivery-permissions.json)
check_command "$POLICY_RESULT"
echo "Custom policy attached"

# Wait for IAM role to propagate
echo "Waiting for IAM role to propagate (15 seconds)..."
sleep 15

# Step 4: Check if configuration recorder already exists
CONFIG_RECORDER_NAME="default"
CREATED_NEW_CONFIG_RECORDER="false"
```

```
echo "Checking for existing configuration recorder..."
EXISTING_RECORDERS=$(aws configservice describe-configuration-recorders 2>/dev/
null || echo "")
if echo "$EXISTING_RECORDERS" | grep -q "name"; then
    echo "Configuration recorder already exists. Will update it."
    # Get the name of the existing recorder
    CONFIG_RECORDER_NAME=$(echo "$EXISTING_RECORDERS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing configuration recorder: $CONFIG_RECORDER_NAME"
else
    echo "No existing configuration recorder found. Will create a new one."
    CREATED_NEW_CONFIG_RECORDER="true"
fi

echo "Creating configuration recorder configuration..."
cat > configurationRecorder.json << EOF
{
  "name": "${CONFIG_RECORDER_NAME}",
  "roleARN": "${ROLE_ARN}",
  "recordingMode": {
    "recordingFrequency": "CONTINUOUS"
  }
}
EOF

echo "Creating recording group configuration..."
cat > recordingGroup.json << EOF
{
  "allSupported": true,
  "includeGlobalResourceTypes": true
}
EOF

echo "Setting up configuration recorder..."
RECORDER_RESULT=$(aws configservice put-configuration-recorder --configuration-
recorder file://configurationRecorder.json --recording-group file://
recordingGroup.json)
check_command "$RECORDER_RESULT"
echo "Configuration recorder set up"

if [ "$CREATED_NEW_CONFIG_RECORDER" = "true" ]; then
    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:config-recorder/${CONFIG_RECORDER_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
```

```
    echo "Tags applied to configuration recorder"
fi

# Step 5: Check if delivery channel already exists
DELIVERY_CHANNEL_NAME="default"
CREATED_NEW_DELIVERY_CHANNEL="false"

echo "Checking for existing delivery channel..."
EXISTING_CHANNELS=$(aws configservice describe-delivery-channels 2>/dev/null ||
 echo "")
if echo "$EXISTING_CHANNELS" | grep -q "name"; then
    echo "Delivery channel already exists."
    # Get the name of the existing channel
    DELIVERY_CHANNEL_NAME=$(echo "$EXISTING_CHANNELS" | grep -o '"name": "[^"]*"'
 | head -1 | cut -d'"' -f4)
    echo "Using existing delivery channel: $DELIVERY_CHANNEL_NAME"

    # Update the existing delivery channel
    echo "Creating delivery channel configuration for update..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Updating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel updated"
else
    echo "No existing delivery channel found. Will create a new one."
    CREATED_NEW_DELIVERY_CHANNEL="true"

    echo "Creating delivery channel configuration..."
    cat > deliveryChannel.json << EOF
{
  "name": "${DELIVERY_CHANNEL_NAME}",
  "s3BucketName": "${S3_BUCKET_NAME}",
```

```
  "snsTopicARN": "${SNS_TOPIC_ARN}",
  "configSnapshotDeliveryProperties": {
    "deliveryFrequency": "Six_Hours"
  }
}
EOF

    echo "Creating delivery channel..."
    CHANNEL_RESULT=$(aws configservice put-delivery-channel --delivery-channel
 file://deliveryChannel.json)
    check_command "$CHANNEL_RESULT"
    echo "Delivery channel created"

    aws configservice tag-resource --resource-arn "arn:aws:config:
${AWS_REGION}:${ACCOUNT_ID}:delivery-channel/${DELIVERY_CHANNEL_NAME}" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=aws-config-gs
    echo "Tags applied to delivery channel"
fi

# Step 6: Start the configuration recorder
echo "Checking configuration recorder status..."
RECORDER_STATUS=$(aws configservice describe-configuration-recorder-status 2>/
dev/null || echo "")
if echo "$RECORDER_STATUS" | grep -q '"recording": true'; then
    echo "Configuration recorder is already running."
else
    echo "Starting configuration recorder..."
    START_RESULT=$(aws configservice start-configuration-recorder --
configuration-recorder-name "$CONFIG_RECORDER_NAME")
    check_command "$START_RESULT"
    echo "Configuration recorder started"
fi

# Step 7: Verify the AWS Config setup
echo "Verifying delivery channel..."
VERIFY_CHANNEL=$(aws configservice describe-delivery-channels)
check_command "$VERIFY_CHANNEL"
echo "$VERIFY_CHANNEL"

echo "Verifying configuration recorder..."
VERIFY_RECORDER=$(aws configservice describe-configuration-recorders)
check_command "$VERIFY_RECORDER"
echo "$VERIFY_RECORDER"
```

```
echo "Verifying configuration recorder status..."
VERIFY_STATUS=$(aws configservice describe-configuration-recorder-status)
check_command "$VERIFY_STATUS"
echo "$VERIFY_STATUS"

# Display created resources
display_resources

# Ask if user wants to clean up resources
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE='y'

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..."
    cleanup_resources
    echo "Cleanup completed."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
fi

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateBucket
  - CreateRole
  - CreateTopic
  - DeleteBucket
  - DeleteConfigurationRecorder
  - DeleteDeliveryChannel
  - DeleteRole
  - DeleteRolePolicy
  - DeleteTopic
  - DescribeConfigurationRecorderStatus

- DescribeConfigurationRecorders
- DescribeDeliveryChannels
- DetachRolePolicy
- GetCallerIdentity
- PutConfigurationRecorder
- PutDeliveryChannel
- PutPublicAccessBlock
- PutRolePolicy
- Rm
- StartConfigurationRecorder
- StopConfigurationRecorder

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de registros de contêineres

O código de exemplo abaixo mostra como:

- Criar uma imagem do Docker.
- Criar um repositório do Amazon ECR.
- Excluir recursos.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash
```

```
# Amazon ECR Getting Started Script
# This script demonstrates the lifecycle of a Docker image in Amazon ECR

# Set up logging
LOG_FILE="ecr-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "Amazon ECR Getting Started Tutorial"
echo "================================================="
echo "This script will:"
echo "1. Create a Docker image"
echo "2. Create an Amazon ECR repository"
echo "3. Authenticate to Amazon ECR"
echo "4. Push the image to Amazon ECR"
echo "5. Pull the image from Amazon ECR"
echo "6. Clean up resources (optional)"
echo "================================================="

# Check prerequisites
echo "Checking prerequisites..."

# Check if AWS CLI is installed
if ! command -v aws &> /dev/null; then
    echo "ERROR: AWS CLI is not installed. Please install it before running this
 script."
    echo "Visit https://docs.aws.amazon.com/cli/latest/userguide/install-
cliv2.html for installation instructions."
    exit 1
fi

# Check if AWS CLI is configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "ERROR: AWS CLI is not configured properly. Please run 'aws configure'
 to set up your credentials."
    exit 1
fi

# Check if Docker is installed
if ! command -v docker &> /dev/null; then
    echo "ERROR: Docker is not installed. Please install Docker before running
 this script."
```

```
    echo "Visit https://docs.docker.com/get-docker/ for installation
 instructions."
    exit 1
fi

# Check if Docker daemon is running
if ! docker info &> /dev/null; then
    echo "ERROR: Docker daemon is not running. Please start Docker and try
 again."
    exit 1
fi

echo "All prerequisites met."

# Initialize variables
REPO_URI=""
TIMEOUT_CMD="timeout 300"  # 5-minute timeout for long-running commands

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Check the log file for details: $LOG_FILE"

    echo "=================================================="
    echo "Resources created:"
    echo "- Docker image: hello-world (local)"
    if [ -n "$REPO_URI" ]; then
        echo "- ECR Repository: hello-repository"
        echo "- ECR Image: $REPO_URI:latest"
    fi
    echo "=================================================="

    echo "Attempting to clean up resources..."
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "=================================================="
    echo "Cleaning up resources..."

    # Delete the image from ECR if it exists
    if [ -n "$REPO_URI" ]; then
```

```
        echo "Deleting image from ECR repository..."
        aws ecr batch-delete-image --repository-name hello-repository --image-ids
 imageTag=latest || echo "Failed to delete image, it may not exist or may have
 already been deleted."
    fi

    # Delete the ECR repository if it exists
    if [ -n "$REPO_URI" ]; then
        echo "Deleting ECR repository..."
        aws ecr delete-repository --repository-name hello-repository --force ||
 echo "Failed to delete repository, it may not exist or may have already been
 deleted."
    fi

    # Remove local Docker image
    echo "Removing local Docker image..."
    docker rmi hello-world:latest 2>/dev/null || echo "Failed to remove local
 image, it may not exist or may have already been deleted."
    if [ -n "$REPO_URI" ]; then
        docker rmi "$REPO_URI:latest" 2>/dev/null || echo "Failed to remove
 tagged image, it may not exist or may have already been deleted."
    fi

    echo "Cleanup completed."
    echo "================================================="
}

# Step 1: Create a Docker image
echo "Step 1: Creating a Docker image"

# Create Dockerfile
echo "Creating Dockerfile..."
cat > Dockerfile << 'EOF'
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Install dependencies
RUN yum update -y && \
 yum install -y httpd

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
```

```
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
EOF

# Build Docker image
echo "Building Docker image..."
$TIMEOUT_CMD docker build -t hello-world . || handle_error "Failed to build
 Docker image or operation timed out after 5 minutes"

# Verify image was created
echo "Verifying Docker image..."
docker images --filter reference=hello-world || handle_error "Failed to list
 Docker images"

echo "Docker image created successfully."

# Step 2: Create an Amazon ECR repository
echo "Step 2: Creating an Amazon ECR repository"

# Get AWS account ID
echo "Getting AWS account ID..."
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ -z "$AWS_ACCOUNT_ID" || "$AWS_ACCOUNT_ID" == *"error"* ]]; then
    handle_error "Failed to get AWS account ID. Make sure your AWS credentials
 are configured correctly."
fi
echo "AWS Account ID: $AWS_ACCOUNT_ID"

# Get current region
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"  # Default to us-east-1 if no region is configured
    echo "No AWS region configured, defaulting to $AWS_REGION"
else
    echo "Using AWS region: $AWS_REGION"
fi

# Create ECR repository
echo "Creating ECR repository..."
```

```
REPO_RESULT=$(aws ecr create-repository --repository-name hello-repository --
tags key=project,value=doc-smith key=tutorial,value=amazon-elastic-container-
registry-gs)
if [[ -z "$REPO_RESULT" || "$REPO_RESULT" == *"error"* ]]; then
    handle_error "Failed to create ECR repository"
fi

# Extract repository URI
REPO_URI="$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/hello-repository"
echo "Repository URI: $REPO_URI"

# Step 3: Authenticate to Amazon ECR
echo "Step 3: Authenticating to Amazon ECR"

echo "Getting ECR login password..."
aws ecr get-login-password --region "$AWS_REGION" | docker login --username
 AWS --password-stdin "$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com" ||
 handle_error "Failed to authenticate to ECR"

echo "Successfully authenticated to ECR."

# Step 4: Push the image to Amazon ECR
echo "Step 4: Pushing the image to Amazon ECR"

# Tag the image
echo "Tagging Docker image..."
docker tag hello-world:latest "$REPO_URI:latest" || handle_error "Failed to tag
 Docker image"

# Push the image with timeout
echo "Pushing image to ECR..."
$TIMEOUT_CMD docker push "$REPO_URI:latest" || handle_error "Failed to push image
 to ECR or operation timed out after 5 minutes"

echo "Successfully pushed image to ECR."

# Step 5: Pull the image from Amazon ECR
echo "Step 5: Pulling the image from Amazon ECR"

# Remove local image to ensure we're pulling from ECR
echo "Removing local tagged image..."
docker rmi "$REPO_URI:latest" || echo "Warning: Failed to remove local tagged
 image"
```

```
# Pull the image with timeout
echo "Pulling image from ECR..."
$TIMEOUT_CMD docker pull "$REPO_URI:latest" || handle_error "Failed to pull image
 from ECR or operation timed out after 5 minutes"

echo "Successfully pulled image from ECR."

# List resources created
echo "================================================="
echo "Resources created:"
echo "- Docker image: hello-world (local)"
echo "- ECR Repository: hello-repository"
echo "- ECR Image: $REPO_URI:latest"
echo "================================================="

# Ask user if they want to clean up resources
echo ""
echo "============================================"
echo "CLEANUP CONFIRMATION"
echo "============================================"
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    # Step 6: Delete the image from ECR
    echo "Step 6: Deleting the image from ECR"

    DELETE_IMAGE_RESULT=$(aws ecr batch-delete-image --repository-name hello-
repository --image-ids imageTag=latest)
    if [[ -z "$DELETE_IMAGE_RESULT" || "$DELETE_IMAGE_RESULT" == *"error"* ]];
 then
        echo "Warning: Failed to delete image from ECR"
    else
        echo "Successfully deleted image from ECR."
    fi

    # Step 7: Delete the ECR repository
    echo "Step 7: Deleting the ECR repository"

    DELETE_REPO_RESULT=$(aws ecr delete-repository --repository-name hello-
repository --force)
    if [[ -z "$DELETE_REPO_RESULT" || "$DELETE_REPO_RESULT" == *"error"* ]]; then
        echo "Warning: Failed to delete ECR repository"
    else
```

```
        echo "Successfully deleted ECR repository."
    fi

    # Remove local Docker images
    echo "Removing local Docker images..."
    docker rmi hello-world:latest 2>/dev/null || echo "Warning: Failed to remove
 local image"

    echo "All resources have been cleaned up."
else
    echo "Resources were not cleaned up. You can manually clean up later with:"
    echo "aws ecr batch-delete-image --repository-name hello-repository --image-
ids imageTag=latest"
    echo "aws ecr delete-repository --repository-name hello-repository --force"
    echo "docker rmi hello-world:latest"
    echo "docker rmi $REPO_URI:latest"
fi

echo "================================================="
echo "Tutorial completed!"
echo "Log file: $LOG_FILE"
echo "================================================="
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - BatchDeleteImage
  - CreateRepository
  - DeleteRepository
  - GetCallerIdentity
  - GetLoginPassword

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de mensagens da Internet das Coisas

O código de exemplo abaixo mostra como:

- Criar recursos de IoT

- Configurar o dispositivo
- Executar os aplicativos de exemplo
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS IoT Core Getting Started Script
# This script creates AWS IoT resources, configures a device, and runs a sample
 application

# Set up logging
LOG_FILE="iot-core-setup.log"
echo "Starting AWS IoT Core setup at $(date)" > $LOG_FILE

# Function to log commands and their outputs
log_cmd() {
    echo "$(date): Running command: $1" >> $LOG_FILE
    eval "$1" 2>&1 | tee -a $LOG_FILE
    return ${PIPESTATUS[0]}
}

# Function to check for errors
check_error() {
    if [ $1 -ne 0 ]; then
        echo "ERROR: Command failed with exit code $1" | tee -a $LOG_FILE
        echo "Please check the log file $LOG_FILE for details" | tee -a $LOG_FILE
        cleanup_on_error
        exit $1
    fi
}
```

```
# Function to cleanup resources on error
cleanup_on_error() {
    echo "Error encountered. Attempting to clean up resources..." | tee -a
 $LOG_FILE
    echo "Resources created:" | tee -a $LOG_FILE
    if [ ! -z "$CERTIFICATE_ARN" ]; then
        echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
        if [ ! -z "$POLICY_NAME" ]; then
            log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
        fi
        if [ ! -z "$THING_NAME" ]; then
            log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --
principal $CERTIFICATE_ARN"
        fi
        if [ ! -z "$CERTIFICATE_ID" ]; then
            log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID
 --new-status INACTIVE"
            log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"
        fi
    fi
    if [ ! -z "$THING_NAME" ]; then
        echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-thing --thing-name $THING_NAME"
    fi
    if [ ! -z "$POLICY_NAME" ]; then
        echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    fi
    if [ ! -z "$SHARED_POLICY_NAME" ]; then
        echo "Shared Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
        log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"
    fi
}

# Generate unique identifiers
RANDOM_SUFFIX=$(openssl rand -hex 4)
THING_NAME="MyIoTThing-${RANDOM_SUFFIX}"
POLICY_NAME="MyIoTPolicy-${RANDOM_SUFFIX}"
SHARED_POLICY_NAME="SharedSubPolicy-${RANDOM_SUFFIX}"
CERTS_DIR="$HOME/certs"

echo "================================================" | tee -a $LOG_FILE
echo "AWS IoT Core Getting Started" | tee -a $LOG_FILE
```

```
echo "================================================" | tee -a $LOG_FILE
echo "This script will:" | tee -a $LOG_FILE
echo "1. Create AWS IoT resources (policy, thing, certificate)" | tee -a
 $LOG_FILE
echo "2. Configure your device" | tee -a $LOG_FILE
echo "3. Set up for running the sample application" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Certificates Directory: $CERTS_DIR" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE

# Get AWS account ID
echo "Getting AWS account ID..." | tee -a $LOG_FILE
ACCOUNT_ID=$(log_cmd "aws sts get-caller-identity --query Account --output text")
check_error $?

# Get AWS region
echo "Getting AWS region..." | tee -a $LOG_FILE
REGION=$(log_cmd "aws configure get region")
check_error $?
if [ -z "$REGION" ]; then
    echo "AWS region not configured. Please run 'aws configure' to set your
 region." | tee -a $LOG_FILE
    exit 1
fi

echo "Using AWS Account ID: $ACCOUNT_ID and Region: $REGION" | tee -a $LOG_FILE

# Step 1: Create AWS IoT Resources
echo "" | tee -a $LOG_FILE
echo "Step 1: Creating AWS IoT Resources..." | tee -a $LOG_FILE

# Create IoT policy
echo "Creating IoT policy document..." | tee -a $LOG_FILE
cat > iot-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
```

```
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/test-*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic"
      ]
    }
  ]
}
EOF

echo "Creating IoT policy: $POLICY_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-policy --policy-name $POLICY_NAME --policy-
document file://iot-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

# Create IoT thing
echo "Creating IoT thing: $THING_NAME..." | tee -a $LOG_FILE
log_cmd "aws iot create-thing --thing-name $THING_NAME"
check_error $?

THING_ARN=$(aws iot describe-thing --thing-name "$THING_NAME" --query 'thingArn'
 --output text)
aws iot tag-resource --resource-arn "$THING_ARN" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-iot-core-gs
```

```
# Create directory for certificates
echo "Creating certificates directory..." | tee -a $LOG_FILE
log_cmd "mkdir -p $CERTS_DIR"
check_error $?

# Create keys and certificate
echo "Creating keys and certificate..." | tee -a $LOG_FILE
CERT_OUTPUT=$(log_cmd "aws iot create-keys-and-certificate --set-as-active
 --certificate-pem-outfile $CERTS_DIR/device.pem.crt --public-key-outfile
 $CERTS_DIR/public.pem.key --private-key-outfile $CERTS_DIR/private.pem.key")
check_error $?

# Extract certificate ARN and ID
CERTIFICATE_ARN=$(echo "$CERT_OUTPUT" | grep "certificateArn" | cut -d'"' -f4)
CERTIFICATE_ID=$(echo "$CERTIFICATE_ARN" | cut -d/ -f2)

if [ -z "$CERTIFICATE_ARN" ] || [ -z "$CERTIFICATE_ID" ]; then
    echo "Failed to extract certificate ARN or ID" | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE

# Attach policy to certificate
echo "Attaching policy to certificate..." | tee -a $LOG_FILE
log_cmd "aws iot attach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Attach certificate to thing
echo "Attaching certificate to thing..." | tee -a $LOG_FILE
log_cmd "aws iot attach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"
check_error $?

# Download Amazon Root CA certificate
echo "Downloading Amazon Root CA certificate..." | tee -a $LOG_FILE
log_cmd "curl -s -o $CERTS_DIR/Amazon-root-CA-1.pem https://www.amazontrust.com/
repository/AmazonRootCA1.pem"
check_error $?

# Step 2: Configure Your Device
```

```
echo "" | tee -a $LOG_FILE
echo "Step 2: Configuring Your Device..." | tee -a $LOG_FILE

# Check if Git is installed
echo "Checking if Git is installed..." | tee -a $LOG_FILE
if ! command -v git &> /dev/null; then
    echo "Git is not installed. Please install Git and run this script again." |
 tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Check if Python is installed
echo "Checking if Python is installed..." | tee -a $LOG_FILE
if ! command -v python3 &> /dev/null; then
    echo "Python 3 is not installed. Please install Python 3 and run this script
 again." | tee -a $LOG_FILE
    cleanup_on_error
    exit 1
fi

# Install AWS IoT Device SDK for Python
echo "Installing AWS IoT Device SDK for Python..." | tee -a $LOG_FILE
log_cmd "python3 -m pip install awsiotsdk"
check_error $?

# Clone the AWS IoT Device SDK for Python repository
echo "Cloning AWS IoT Device SDK for Python repository..." | tee -a $LOG_FILE
if [ ! -d "$HOME/aws-iot-device-sdk-python-v2" ]; then
    log_cmd "cd $HOME && git clone https://github.com/aws/aws-iot-device-sdk-
python-v2.git"
    check_error $?
else
    echo "AWS IoT Device SDK for Python repository already exists." | tee -a
 $LOG_FILE
fi

# Step 3: Get AWS IoT Endpoint
echo "" | tee -a $LOG_FILE
echo "Step 3: Getting AWS IoT Endpoint..." | tee -a $LOG_FILE

IOT_ENDPOINT=$(log_cmd "aws iot describe-endpoint --endpoint-type iot:Data-ATS --
query endpointAddress --output text")
check_error $?
```

```
echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE

# Create a shared subscription policy (optional)
echo "" | tee -a $LOG_FILE
echo "Creating shared subscription policy (optional)..." | tee -a $LOG_FILE

cat > shared-sub-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:client/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topic/test/topic"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/test/topic",
        "arn:aws:iot:$REGION:$ACCOUNT_ID:topicfilter/\$share/*/test/topic"
      ]
    }
  ]
}
EOF
```

```
log_cmd "aws iot create-policy --policy-name $SHARED_POLICY_NAME --policy-
document file://shared-sub-policy.json --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-iot-core-gs"
check_error $?

log_cmd "aws iot attach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"
check_error $?

# Summary of created resources
echo "" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Setup Complete! Resources Created:" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Thing Name: $THING_NAME" | tee -a $LOG_FILE
echo "Policy Name: $POLICY_NAME" | tee -a $LOG_FILE
echo "Shared Subscription Policy Name: $SHARED_POLICY_NAME" | tee -a $LOG_FILE
echo "Certificate ID: $CERTIFICATE_ID" | tee -a $LOG_FILE
echo "Certificate ARN: $CERTIFICATE_ARN" | tee -a $LOG_FILE
echo "Certificate Files Location: $CERTS_DIR" | tee -a $LOG_FILE
echo "AWS IoT Endpoint: $IOT_ENDPOINT" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE

# Instructions for running the sample application
echo "" | tee -a $LOG_FILE
echo "To run the sample application, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 pubsub.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key" | tee -a $LOG_FILE
echo "" | tee -a $LOG_FILE
echo "To run the shared subscription example, execute:" | tee -a $LOG_FILE
echo "cd $HOME/aws-iot-device-sdk-python-v2/samples" | tee -a $LOG_FILE
echo "python3 mqtt5_shared_subscription.py \\" | tee -a $LOG_FILE
echo "  --endpoint $IOT_ENDPOINT \\" | tee -a $LOG_FILE
echo "  --ca_file $CERTS_DIR/Amazon-root-CA-1.pem \\" | tee -a $LOG_FILE
echo "  --cert $CERTS_DIR/device.pem.crt \\" | tee -a $LOG_FILE
echo "  --key $CERTS_DIR/private.pem.key \\" | tee -a $LOG_FILE
echo "  --group_identifier consumer" | tee -a $LOG_FILE

# Ask if user wants to clean up resources
echo "" | tee -a $LOG_FILE
```

```
echo "================================================" | tee -a $LOG_FILE
echo "CLEANUP CONFIRMATION" | tee -a $LOG_FILE
echo "================================================" | tee -a $LOG_FILE
echo "Do you want to clean up all created resources? (y/n): " | tee -a $LOG_FILE
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a $LOG_FILE

    # Detach policies from certificate
    echo "Detaching policies from certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN"
    log_cmd "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN"

    # Detach certificate from thing
    echo "Detaching certificate from thing..." | tee -a $LOG_FILE
    log_cmd "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN"

    # Update certificate status to INACTIVE
    echo "Setting certificate to inactive..." | tee -a $LOG_FILE
    log_cmd "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE"

    # Delete certificate
    echo "Deleting certificate..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-certificate --certificate-id $CERTIFICATE_ID"

    # Delete thing
    echo "Deleting thing..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-thing --thing-name $THING_NAME"

    # Delete policies
    echo "Deleting policies..." | tee -a $LOG_FILE
    log_cmd "aws iot delete-policy --policy-name $POLICY_NAME"
    log_cmd "aws iot delete-policy --policy-name $SHARED_POLICY_NAME"

    echo "Cleanup complete!" | tee -a $LOG_FILE
else
    echo "Resources were not cleaned up. You can manually clean them up later." |
 tee -a $LOG_FILE
    echo "To clean up resources, run the following commands:" | tee -a $LOG_FILE
```

```
    echo "aws iot detach-policy --policy-name $POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-policy --policy-name $SHARED_POLICY_NAME --target
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot detach-thing-principal --thing-name $THING_NAME --principal
 $CERTIFICATE_ARN" | tee -a $LOG_FILE
    echo "aws iot update-certificate --certificate-id $CERTIFICATE_ID --new-
status INACTIVE" | tee -a $LOG_FILE
    echo "aws iot delete-certificate --certificate-id $CERTIFICATE_ID" | tee -a
 $LOG_FILE
    echo "aws iot delete-thing --thing-name $THING_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $POLICY_NAME" | tee -a $LOG_FILE
    echo "aws iot delete-policy --policy-name $SHARED_POLICY_NAME" | tee -a
 $LOG_FILE
fi

echo "" | tee -a $LOG_FILE
echo "Script execution completed. See $LOG_FILE for details." | tee -a $LOG_FILE
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachPolicy
  - AttachThingPrincipal
  - CreateKeysAndCertificate
  - CreatePolicy
  - CreateThing
  - DeleteCertificate
  - DeletePolicy
  - DeleteThing
  - DescribeEndpoint
  - DetachPolicy
  - DetachThingPrincipal
  - GetCallerIdentity
  - UpdateCertificate

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de arquivos de atributos de machine learning

O código de exemplo abaixo mostra como:

- Configurar permissões do IAM.
- Criar um perfil de execução do SageMaker.
- Criar grupo de atributos.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon SageMaker Feature Store Tutorial Script - Version 3
# This script demonstrates how to use Amazon SageMaker Feature Store with AWS CLI

# Setup logging
LOG_FILE="sagemaker-featurestore-tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting SageMaker Feature Store tutorial script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"
echo ""

# Track created resources for cleanup
CREATED_RESOURCES=()

# Function to handle errors
```

```
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command status
check_status() {
    if echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to wait for feature group to be created
wait_for_feature_group() {
    local feature_group_name=$1
    local status="Creating"

    echo "Waiting for feature group ${feature_group_name} to be created..."

    while [ "$status" = "Creating" ]; do
        sleep 5
        status=$(aws sagemaker describe-feature-group \
            --feature-group-name "${feature_group_name}" \
            --query 'FeatureGroupStatus' \
            --output text)
        echo "Current status: ${status}"

        if [ "$status" = "Failed" ]; then
            handle_error "Feature group ${feature_group_name} creation failed"
        fi
    done

    echo "Feature group ${feature_group_name} is now ${status}"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Clean up in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        resource="${CREATED_RESOURCES[$i]}"
```

```
        resource_type=$(echo "$resource" | cut -d: -f1)
        resource_name=$(echo "$resource" | cut -d: -f2)

        echo "Deleting $resource_type: $resource_name"

        case "$resource_type" in
            "FeatureGroup")
                aws sagemaker delete-feature-group --feature-group-name
 "$resource_name"
                ;;
            "S3Bucket")
                echo "Emptying S3 bucket: $resource_name"
                aws s3 rm "s3://$resource_name" --recursive 2>/dev/null
                echo "Deleting S3 bucket: $resource_name"
                aws s3api delete-bucket --bucket "$resource_name" 2>/dev/null
                ;;
            "IAMRole")
                echo "Detaching policies from role: $resource_name"
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>/dev/null
                aws iam detach-role-policy --role-name "$resource_name" --policy-
arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>/dev/null
                echo "Deleting IAM role: $resource_name"
                aws iam delete-role --role-name "$resource_name" 2>/dev/null
                ;;
            *)
                echo "Unknown resource type: $resource_type"
                ;;
        esac
    done
}

# Function to create SageMaker execution role
create_sagemaker_role() {
    local role_name="SageMakerFeatureStoreRole-$(openssl rand -hex 4)"

    echo "Creating SageMaker execution role: $role_name" >&2

    # Create trust policy document
    local trust_policy='{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
```

```
                "Principal": {
                    "Service": "sagemaker.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'

    # Create the role
    local role_result=$(aws iam create-role \
        --role-name "$role_name" \
        --assume-role-policy-document "$trust_policy" \
        --description "SageMaker execution role for Feature Store tutorial" 2>&1)

    if echo "$role_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to create IAM role: $role_result"
    fi

    echo "Role created successfully" >&2
    CREATED_RESOURCES+=("IAMRole:$role_name")

    # Tag the role
    echo "Tagging IAM role..." >&2
    aws iam tag-role --role-name "$role_name" --tags Key=project,Value=doc-smith
Key=tutorial,Value=sagemaker-featurestore 2>&1

    # Attach necessary policies
    echo "Attaching policies to role..." >&2

    # SageMaker execution policy
    local policy1_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess" 2>&1)

    if echo "$policy1_result" | grep -i "error" > /dev/null; then
        handle_error "Failed to attach SageMaker policy: $policy1_result"
    fi

    # S3 access policy
    local policy2_result=$(aws iam attach-role-policy \
        --role-name "$role_name" \
        --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess" 2>&1)

    if echo "$policy2_result" | grep -i "error" > /dev/null; then
```

```
        handle_error "Failed to attach S3 policy: $policy2_result"
    fi

    # Get account ID for role ARN
    local account_id=$(aws sts get-caller-identity --query Account --output text)
    local role_arn="arn:aws:iam::${account_id}:role/${role_name}"

    echo "Role ARN: $role_arn" >&2
    echo "Waiting 10 seconds for role to propagate..." >&2
    sleep 10

    # Return only the role ARN to stdout
    echo "$role_arn"
}

# Handle SageMaker execution role
ROLE_ARN=""

if [ -z "$1" ]; then
    echo "Creating SageMaker execution role automatically..."
    ROLE_ARN=$(create_sagemaker_role)
    if [ -z "$ROLE_ARN" ]; then
        handle_error "Failed to create SageMaker execution role"
    fi
else
    ROLE_ARN="$1"

    # Validate the role ARN
    ROLE_NAME=$(echo "$ROLE_ARN" | sed 's/.*role\///')
    ROLE_CHECK=$(aws iam get-role --role-name "$ROLE_NAME" 2>&1)
    if echo "$ROLE_CHECK" | grep -i "error" > /dev/null; then
        echo "Creating a new role automatically..."
        ROLE_ARN=$(create_sagemaker_role)
        if [ -z "$ROLE_ARN" ]; then
            handle_error "Failed to create SageMaker execution role"
        fi
    fi
fi

# Handle cleanup option
AUTO_CLEANUP=""
if [ -n "$2" ]; then
    AUTO_CLEANUP="$2"
fi
```

```
# Generate a random identifier for resource names
RANDOM_ID=$(openssl rand -hex 4)
echo "Using random identifier: $RANDOM_ID"

# Set variables
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
check_status "$ACCOUNT_ID"
echo "Account ID: $ACCOUNT_ID"

# Get current region
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"
    echo "No default region configured, using: $REGION"
else
    echo "Using region: $REGION"
fi
# Check for shared prereq bucket
PREREQ_BUCKET=$(aws cloudformation describe-stacks --stack-name tutorial-prereqs-
bucket \
    --query 'Stacks[0].Outputs[?OutputKey==`BucketName`].OutputValue' --output
 text 2>/dev/null)
if [ -n "$PREREQ_BUCKET" ] && [ "$PREREQ_BUCKET" != "None" ]; then
    S3_BUCKET_NAME="$PREREQ_BUCKET"
    BUCKET_IS_SHARED=true
    echo "Using shared bucket: $S3_BUCKET_NAME"
else
    BUCKET_IS_SHARED=false
    S3_BUCKET_NAME="sagemaker-featurestore-${RANDOM_ID}-${ACCOUNT_ID}"
fi
PREFIX="featurestore-tutorial"
CURRENT_TIME=$(date +%s)

echo "Creating S3 bucket: $S3_BUCKET_NAME"
# Create bucket in current region (skip if using shared bucket)
if [ "$BUCKET_IS_SHARED" = "false" ]; then
if [ "$REGION" = "us-east-1" ]; then
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" 2>&1)
else
    BUCKET_RESULT=$(aws s3api create-bucket --bucket "$S3_BUCKET_NAME" \
        --region "$REGION" \
        --create-bucket-configuration LocationConstraint="$REGION" 2>&1)
```

```
fi

if echo "$BUCKET_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to create S3 bucket: $BUCKET_RESULT"
    exit 1
fi

echo "$BUCKET_RESULT"
CREATED_RESOURCES+=("S3Bucket:$S3_BUCKET_NAME")

# Tag the S3 bucket
echo "Tagging S3 bucket: $S3_BUCKET_NAME"
aws s3api put-bucket-tagging --bucket "$S3_BUCKET_NAME" --tagging
 'TagSet=[{Key=project,Value=doc-smith},{Key=tutorial,Value=sagemaker-
featurestore}]' 2>&1

# Block public access to the bucket
BLOCK_RESULT=$(aws s3api put-public-access-block \
    --bucket "$S3_BUCKET_NAME" \
    --public-access-block-configuration
 "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets
 2>&1)

if echo "$BLOCK_RESULT" | grep -i "error" > /dev/null; then
    echo "Failed to block public access to S3 bucket: $BLOCK_RESULT"
    cleanup_resources
    exit 1
fi
else
    echo "Using shared bucket (skipping creation)"
fi

# Create feature groups
echo "Creating feature groups..."

# Create customers feature group
CUSTOMERS_FEATURE_GROUP_NAME="customers-feature-group-${RANDOM_ID}"
echo "Creating customers feature group: $CUSTOMERS_FEATURE_GROUP_NAME"

CUSTOMERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
```

```
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "name", "FeatureType": "String"},
        {"FeatureName": "age", "FeatureType": "Integral"},
        {"FeatureName": "address", "FeatureType": "String"},
        {"FeatureName": "membership_type", "FeatureType": "String"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$CUSTOMERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create customers feature group: $CUSTOMERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$CUSTOMERS_FEATURE_GROUP_NAME")

# Create orders feature group
ORDERS_FEATURE_GROUP_NAME="orders-feature-group-${RANDOM_ID}"
echo "Creating orders feature group: $ORDERS_FEATURE_GROUP_NAME"

ORDERS_RESPONSE=$(aws sagemaker create-feature-group \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record-identifier-feature-name "customer_id" \
    --event-time-feature-name "EventTime" \
    --feature-definitions '[
        {"FeatureName": "customer_id", "FeatureType": "Integral"},
        {"FeatureName": "order_id", "FeatureType": "String"},
        {"FeatureName": "order_date", "FeatureType": "String"},
        {"FeatureName": "product", "FeatureType": "String"},
        {"FeatureName": "quantity", "FeatureType": "Integral"},
        {"FeatureName": "amount", "FeatureType": "Fractional"},
        {"FeatureName": "EventTime", "FeatureType": "Fractional"}
    ]' \
```

```
    --online-store-config '{"EnableOnlineStore": true}' \
    --offline-store-config '{
        "S3StorageConfig": {
            "S3Uri": "s3://'${S3_BUCKET_NAME}'/'${PREFIX}'"
        },
        "DisableGlueTableCreation": false
    }' \
    --role-arn "$ROLE_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=sagemaker-featurestore
 2>&1)

if echo "$ORDERS_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to create orders feature group: $ORDERS_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDERS_RESPONSE"
CREATED_RESOURCES+=("FeatureGroup:$ORDERS_FEATURE_GROUP_NAME")

# Wait for feature groups to be created
wait_for_feature_group "$CUSTOMERS_FEATURE_GROUP_NAME"
wait_for_feature_group "$ORDERS_FEATURE_GROUP_NAME"

# Ingest data into feature groups
echo "Ingesting data into feature groups..."

# Ingest customer data
echo "Ingesting customer data..."
CUSTOMER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "name", "ValueAsString": "John Doe"},
        {"FeatureName": "age", "ValueAsString": "35"},
        {"FeatureName": "address", "ValueAsString": "123 Main St"},
        {"FeatureName": "membership_type", "ValueAsString": "premium"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 1 data: $CUSTOMER1_RESPONSE"
    cleanup_resources
    exit 1
```

```
fi

echo "$CUSTOMER1_RESPONSE"

CUSTOMER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "name", "ValueAsString": "Jane Smith"},
        {"FeatureName": "age", "ValueAsString": "28"},
        {"FeatureName": "address", "ValueAsString": "456 Oak Ave"},
        {"FeatureName": "membership_type", "ValueAsString": "standard"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$CUSTOMER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest customer 2 data: $CUSTOMER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER2_RESPONSE"

# Ingest order data
echo "Ingesting order data..."
ORDER1_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "573291"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-001"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-15"},
        {"FeatureName": "product", "ValueAsString": "Laptop"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "1299.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER1_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 1 data: $ORDER1_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER1_RESPONSE"
```

```
ORDER2_RESPONSE=$(aws sagemaker-featurestore-runtime put-record \
    --feature-group-name "$ORDERS_FEATURE_GROUP_NAME" \
    --record '[
        {"FeatureName": "customer_id", "ValueAsString": "109382"},
        {"FeatureName": "order_id", "ValueAsString": "ORD-002"},
        {"FeatureName": "order_date", "ValueAsString": "2023-01-20"},
        {"FeatureName": "product", "ValueAsString": "Smartphone"},
        {"FeatureName": "quantity", "ValueAsString": "1"},
        {"FeatureName": "amount", "ValueAsString": "899.99"},
        {"FeatureName": "EventTime", "ValueAsString": "'${CURRENT_TIME}'"}
    ]' 2>&1)

if echo "$ORDER2_RESPONSE" | grep -i "error" > /dev/null; then
    echo "Failed to ingest order 2 data: $ORDER2_RESPONSE"
    cleanup_resources
    exit 1
fi

echo "$ORDER2_RESPONSE"

# Retrieve records from feature groups
echo "Retrieving records from feature groups..."

# Get a single customer record
echo "Getting customer record with ID 573291:"
CUSTOMER_RECORD=$(aws sagemaker-featurestore-runtime get-record \
    --feature-group-name "$CUSTOMERS_FEATURE_GROUP_NAME" \
    --record-identifier-value-as-string "573291" 2>&1)

if echo "$CUSTOMER_RECORD" | grep -i "error" > /dev/null; then
    echo "Failed to get customer record: $CUSTOMER_RECORD"
    cleanup_resources
    exit 1
fi

echo "$CUSTOMER_RECORD"

# Get multiple records using batch-get-record
echo "Getting multiple records using batch-get-record:"
BATCH_RECORDS=$(aws sagemaker-featurestore-runtime batch-get-record \
    --identifiers '[
        {
            "FeatureGroupName": "'${CUSTOMERS_FEATURE_GROUP_NAME}'",
```

```
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        },
        {
            "FeatureGroupName": "'${ORDERS_FEATURE_GROUP_NAME}'",
            "RecordIdentifiersValueAsString": ["573291", "109382"]
        }
    ]' 2>&1)

if echo "$BATCH_RECORDS" | grep -i "error" > /dev/null && ! echo "$BATCH_RECORDS"
 | grep -i "Records" > /dev/null; then
    echo "Failed to get batch records: $BATCH_RECORDS"
    cleanup_resources
    exit 1
fi

echo "$BATCH_RECORDS"

# List feature groups
echo "Listing feature groups:"
FEATURE_GROUPS=$(aws sagemaker list-feature-groups 2>&1)

if echo "$FEATURE_GROUPS" | grep -i "error" > /dev/null; then
    echo "Failed to list feature groups: $FEATURE_GROUPS"
    cleanup_resources
    exit 1
fi

echo "$FEATURE_GROUPS"

# Display summary of created resources
echo ""
echo "==========================================="
echo "TUTORIAL COMPLETED SUCCESSFULLY!"
echo "==========================================="
echo "Resources created:"
echo "- S3 Bucket: $S3_BUCKET_NAME"
echo "- Customers Feature Group: $CUSTOMERS_FEATURE_GROUP_NAME"
echo "- Orders Feature Group: $ORDERS_FEATURE_GROUP_NAME"
if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
    echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o 'IAMRole:
[^[:space:]]*' | cut -d: -f2)"
fi
echo ""
echo "You can now:"
```

```
echo "1. View your feature groups in the SageMaker console"
echo "2. Query the offline store using Amazon Athena"
echo "3. Use the feature groups in your ML workflows"
echo "========================================="
echo ""

# Handle cleanup
if [ "$AUTO_CLEANUP" = "y" ]; then
    echo "Auto-cleanup enabled. Starting cleanup..."
    cleanup_resources
    echo "Cleanup completed."
elif [ "$AUTO_CLEANUP" = "n" ]; then
    echo "Auto-cleanup disabled. Resources will remain in your account."
    echo "To clean up later, run this script again with cleanup option 'y'"
else
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Do you want to clean up all created resources? (y/n): "
    read -r CLEANUP_CHOICE

    if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
        echo "Starting cleanup..."
        cleanup_resources
        echo "Cleanup completed."
    else
        echo "Skipping cleanup. Resources will remain in your account."
        echo "To clean up later, delete the following resources:"
        echo "- Feature Groups: $CUSTOMERS_FEATURE_GROUP_NAME,
 $ORDERS_FEATURE_GROUP_NAME"
        echo "- S3 Bucket: $S3_BUCKET_NAME"
        if [[ " ${CREATED_RESOURCES[@]} " =~ " IAMRole:" ]]; then
            echo "- IAM Role: $(echo "${CREATED_RESOURCES[@]}" | grep -o
 'IAMRole:[^[:space:]]*' | cut -d: -f2)"
        fi
        echo ""
        echo "Estimated ongoing cost: ~$0.01 per month for online store"
    fi
fi

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.

- AttachRolePolicy
- CreateBucket
- CreateFeatureGroup
- CreateRole
- DeleteBucket
- DeleteFeatureGroup
- DeleteRole
- DescribeFeatureGroup
- DetachRolePolicy
- GetCallerIdentity
- GetRole
- ListFeatureGroups
- PutPublicAccessBlock
- Rm

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de clusters gerenciados de kubernetes

O código de exemplo abaixo mostra como:

- Criar uma VPC para um cluster do EKS.
- Criar perfis do IAM para um cluster do EKS.
- Criação do seu cluster de EKS
- Configurar o kubectl para se comunicar com o cluster.
- Crie um grupo de nós gerenciados
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon EKS Cluster Creation Script (v2)
# This script creates an Amazon EKS cluster with a managed node group using the
 AWS CLI

# Set up logging
LOG_FILE="eks-cluster-creation-v2.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon EKS cluster creation script at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check command success
check_command() {
    if [ $? -ne 0 ] || echo "$1" | grep -i "error" > /dev/null; then
        handle_error "$1"
    fi
}

# Function to check if kubectl is installed
check_kubectl() {
    if ! command -v kubectl &> /dev/null; then
        echo "WARNING: kubectl is not installed or not in your PATH."
```

```
        echo ""
        echo "To install kubectl, follow these instructions based on your
 operating system:"
        echo ""
        echo "For Linux:"
        echo "  1. Download the latest release:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl\""
        echo ""
        echo "  2. Make the kubectl binary executable:"
        echo "     chmod +x ./kubectl"
        echo ""
        echo "  3. Move the binary to your PATH:"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For macOS:"
        echo "  1. Using Homebrew:"
        echo "     brew install kubectl"
        echo "     or"
        echo "  2. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/\$(curl -L -s https://
dl.k8s.io/release/stable.txt)/bin/darwin/amd64/kubectl\""
        echo "     chmod +x ./kubectl"
        echo "     sudo mv ./kubectl /usr/local/bin/kubectl"
        echo ""
        echo "For Windows:"
        echo "  1. Using curl:"
        echo "     curl -LO \"https://dl.k8s.io/release/v1.28.0/bin/windows/
amd64/kubectl.exe\""
        echo "     Add the binary to your PATH"
        echo "     or"
        echo "  2. Using Chocolatey:"
        echo "     choco install kubernetes-cli"
        echo ""
        echo "After installation, verify with: kubectl version --client"
        echo ""
        return 1
    fi
    return 0
}

# Generate a random identifier for resource names
RANDOM_ID=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 6 | head -n 1)
STACK_NAME="eks-vpc-stack-${RANDOM_ID}"
```

```
CLUSTER_NAME="eks-cluster-${RANDOM_ID}"
NODEGROUP_NAME="eks-nodegroup-${RANDOM_ID}"
CLUSTER_ROLE_NAME="EKSClusterRole-${RANDOM_ID}"
NODE_ROLE_NAME="EKSNodeRole-${RANDOM_ID}"

echo "Using the following resource names:"
echo "- VPC Stack: $STACK_NAME"
echo "- EKS Cluster: $CLUSTER_NAME"
echo "- Node Group: $NODEGROUP_NAME"
echo "- Cluster IAM Role: $CLUSTER_ROLE_NAME"
echo "- Node IAM Role: $NODE_ROLE_NAME"

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources in reverse order..."

    # Check if node group exists and delete it
    if aws eks list-nodegroups --cluster-name "$CLUSTER_NAME" --query
 "nodegroups[?contains(@,'$NODEGROUP_NAME')]" --output text 2>/dev/null | grep -q
 "$NODEGROUP_NAME"; then
        echo "Deleting node group: $NODEGROUP_NAME"
        aws eks delete-nodegroup --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
        echo "Waiting for node group deletion to complete..."
        aws eks wait nodegroup-deleted --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME"
        echo "Node group deleted successfully."
    fi

    # Check if cluster exists and delete it
    if aws eks describe-cluster --name "$CLUSTER_NAME" 2>/dev/null; then
        echo "Deleting cluster: $CLUSTER_NAME"
        aws eks delete-cluster --name "$CLUSTER_NAME"
        echo "Waiting for cluster deletion to complete (this may take several
 minutes)..."
        aws eks wait cluster-deleted --name "$CLUSTER_NAME"
        echo "Cluster deleted successfully."
    fi

    # Check if CloudFormation stack exists and delete it
```

```
    if aws cloudformation describe-stacks --stack-name "$STACK_NAME" 2>/dev/null;
 then
        echo "Deleting CloudFormation stack: $STACK_NAME"
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "Waiting for CloudFormation stack deletion to complete..."
        aws cloudformation wait stack-delete-complete --stack-name "$STACK_NAME"
        echo "CloudFormation stack deleted successfully."
    fi

    # Clean up IAM roles
    if aws iam get-role --role-name "$NODE_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from node role: $NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly --role-name "$NODE_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKS_CNI_Policy --role-name "$NODE_ROLE_NAME"
        echo "Deleting node role: $NODE_ROLE_NAME"
        aws iam delete-role --role-name "$NODE_ROLE_NAME"
        echo "Node role deleted successfully."
    fi

    if aws iam get-role --role-name "$CLUSTER_ROLE_NAME" 2>/dev/null; then
        echo "Detaching policies from cluster role: $CLUSTER_ROLE_NAME"
        aws iam detach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name "$CLUSTER_ROLE_NAME"
        echo "Deleting cluster role: $CLUSTER_ROLE_NAME"
        aws iam delete-role --role-name "$CLUSTER_ROLE_NAME"
        echo "Cluster role deleted successfully."
    fi

    echo "Cleanup complete."
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Cleaning up resources..."; cleanup_resources;
 exit 1' SIGINT SIGTERM

# Verify AWS CLI configuration
echo "Verifying AWS CLI configuration..."
AWS_ACCOUNT_INFO=$(aws sts get-caller-identity)
check_command "$AWS_ACCOUNT_INFO"
echo "AWS CLI is properly configured."
```

```
# Step 1: Create VPC using CloudFormation
echo "Step 1: Creating VPC with CloudFormation..."
echo "Creating CloudFormation stack: $STACK_NAME"

# Create the CloudFormation stack
CF_CREATE_OUTPUT=$(aws cloudformation create-stack \
  --stack-name "$STACK_NAME" \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-eks/
cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs)
check_command "$CF_CREATE_OUTPUT"
CREATED_RESOURCES+=("CloudFormation Stack: $STACK_NAME")

echo "Waiting for CloudFormation stack to complete (this may take a few
 minutes)..."
aws cloudformation wait stack-create-complete --stack-name "$STACK_NAME"
if [ $? -ne 0 ]; then
    handle_error "CloudFormation stack creation failed"
fi
echo "CloudFormation stack created successfully."

# Step 2: Create IAM roles for EKS
echo "Step 2: Creating IAM roles for EKS..."

# Create cluster role trust policy
echo "Creating cluster role trust policy..."
cat > eks-cluster-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create cluster role
echo "Creating cluster IAM role: $CLUSTER_ROLE_NAME"
```

```
CLUSTER_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$CLUSTER_ROLE_NAME" \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json")
check_command "$CLUSTER_ROLE_OUTPUT"
aws iam tag-role --role-name "$CLUSTER_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $CLUSTER_ROLE_NAME")

# Attach policy to cluster role
echo "Attaching EKS cluster policy to role..."
ATTACH_CLUSTER_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name "$CLUSTER_ROLE_NAME")
check_command "$ATTACH_CLUSTER_POLICY_OUTPUT"

# Create node role trust policy
echo "Creating node role trust policy..."
cat > node-role-trust-policy.json << EOF
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create node role
echo "Creating node IAM role: $NODE_ROLE_NAME"
NODE_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$NODE_ROLE_NAME" \
  --assume-role-policy-document file://"node-role-trust-policy.json")
check_command "$NODE_ROLE_OUTPUT"
aws iam tag-role --role-name "$NODE_ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=eks-gs
CREATED_RESOURCES+=("IAM Role: $NODE_ROLE_NAME")

# Attach policies to node role
echo "Attaching EKS node policies to role..."
```

```
ATTACH_NODE_POLICY1_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY1_OUTPUT"

ATTACH_NODE_POLICY2_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY2_OUTPUT"

ATTACH_NODE_POLICY3_OUTPUT=$(aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name "$NODE_ROLE_NAME")
check_command "$ATTACH_NODE_POLICY3_OUTPUT"

# Step 3: Get VPC and subnet information
echo "Step 3: Getting VPC and subnet information..."

VPC_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='VpcId'].OutputValue" \
  --output text)
if [ -z "$VPC_ID" ]; then
    handle_error "Failed to get VPC ID from CloudFormation stack"
fi
echo "VPC ID: $VPC_ID"

SUBNET_IDS=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SubnetIds'].OutputValue" \
  --output text)
if [ -z "$SUBNET_IDS" ]; then
    handle_error "Failed to get Subnet IDs from CloudFormation stack"
fi
echo "Subnet IDs: $SUBNET_IDS"

SECURITY_GROUP_ID=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query "Stacks[0].Outputs[?OutputKey=='SecurityGroups'].OutputValue" \
  --output text)
if [ -z "$SECURITY_GROUP_ID" ]; then
    handle_error "Failed to get Security Group ID from CloudFormation stack"
fi
echo "Security Group ID: $SECURITY_GROUP_ID"
```

```
# Step 4: Create EKS cluster
echo "Step 4: Creating EKS cluster: $CLUSTER_NAME"

CLUSTER_ROLE_ARN=$(aws iam get-role --role-name "$CLUSTER_ROLE_NAME" --query
 "Role.Arn" --output text)
if [ -z "$CLUSTER_ROLE_ARN" ]; then
    handle_error "Failed to get Cluster Role ARN"
fi

echo "Creating EKS cluster (this will take 10-15 minutes)..."
CREATE_CLUSTER_OUTPUT=$(aws eks create-cluster \
  --name "$CLUSTER_NAME" \
  --role-arn "$CLUSTER_ROLE_ARN" \
  --resources-vpc-config
 subnetIds="$SUBNET_IDS",securityGroupIds="$SECURITY_GROUP_ID" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_CLUSTER_OUTPUT"
CREATED_RESOURCES+=("EKS Cluster: $CLUSTER_NAME")

echo "Waiting for EKS cluster to become active (this may take 10-15 minutes)..."
aws eks wait cluster-active --name "$CLUSTER_NAME"
if [ $? -ne 0 ]; then
    handle_error "Cluster creation failed or timed out"
fi
echo "EKS cluster is now active."

# Step 5: Configure kubectl
echo "Step 5: Configuring kubectl to communicate with the cluster..."

# Check if kubectl is installed
if ! check_kubectl; then
    echo "Will skip kubectl configuration steps but continue with the script."
    echo "You can manually configure kubectl later with: aws eks update-
kubeconfig --name \"$CLUSTER_NAME\""
else
    UPDATE_KUBECONFIG_OUTPUT=$(aws eks update-kubeconfig --name "$CLUSTER_NAME")
    check_command "$UPDATE_KUBECONFIG_OUTPUT"
    echo "kubectl configured successfully."

    # Test kubectl configuration
    echo "Testing kubectl configuration..."
    KUBECTL_TEST_OUTPUT=$(kubectl get svc 2>&1)
    if [ $? -ne 0 ]; then
```

```
        echo "Warning: kubectl configuration test failed. This might be due to
 permissions or network issues."
        echo "Error details: $KUBECTL_TEST_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$KUBECTL_TEST_OUTPUT"
        echo "kubectl configuration test successful."
    fi
fi

# Step 6: Create managed node group
echo "Step 6: Creating managed node group: $NODEGROUP_NAME"

NODE_ROLE_ARN=$(aws iam get-role --role-name "$NODE_ROLE_NAME" --query "Role.Arn"
 --output text)
if [ -z "$NODE_ROLE_ARN" ]; then
    handle_error "Failed to get Node Role ARN"
fi

# Convert comma-separated subnet IDs to space-separated for the create-nodegroup
 command
SUBNET_IDS_ARRAY=(${SUBNET_IDS//,/ })

echo "Creating managed node group (this will take 5-10 minutes)..."
CREATE_NODEGROUP_OUTPUT=$(aws eks create-nodegroup \
  --cluster-name "$CLUSTER_NAME" \
  --nodegroup-name "$NODEGROUP_NAME" \
  --node-role "$NODE_ROLE_ARN" \
  --subnets "${SUBNET_IDS_ARRAY[@]}" \
  --tags Key=project,Value=doc-smith,Key=tutorial,Value=eks-gs)
check_command "$CREATE_NODEGROUP_OUTPUT"
CREATED_RESOURCES+=("EKS Node Group: $NODEGROUP_NAME")

echo "Waiting for node group to become active (this may take 5-10 minutes)..."
aws eks wait nodegroup-active --cluster-name "$CLUSTER_NAME" --nodegroup-name
 "$NODEGROUP_NAME"
if [ $? -ne 0 ]; then
    handle_error "Node group creation failed or timed out"
fi
echo "Node group is now active."

# Step 7: Verify nodes
echo "Step 7: Verifying nodes..."
```

```
echo "Waiting for nodes to register with the cluster (this may take a few
 minutes)..."
sleep 60  # Give nodes more time to register

# Check if kubectl is installed before attempting to use it
if ! check_kubectl; then
    echo "Cannot verify nodes without kubectl. Skipping this step."
    echo "You can manually verify nodes after installing kubectl with: kubectl
 get nodes"
else
    NODES_OUTPUT=$(kubectl get nodes 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get nodes. This might be due to permissions or
 the nodes are still registering."
        echo "Error details: $NODES_OUTPUT"
        echo "Continuing with script execution..."
    else
        echo "$NODES_OUTPUT"
        echo "Nodes verified successfully."
    fi
fi

# Step 8: View resources
echo "Step 8: Viewing cluster resources..."

echo "Cluster information:"
CLUSTER_INFO=$(aws eks describe-cluster --name "$CLUSTER_NAME")
echo "$CLUSTER_INFO"

echo "Node group information:"
NODEGROUP_INFO=$(aws eks describe-nodegroup --cluster-name "$CLUSTER_NAME" --
nodegroup-name "$NODEGROUP_NAME")
echo "$NODEGROUP_INFO"

echo "Kubernetes resources:"
if ! check_kubectl; then
    echo "Cannot list Kubernetes resources without kubectl. Skipping this step."
    echo "You can manually list resources after installing kubectl with: kubectl
 get all --all-namespaces"
else
    KUBE_RESOURCES=$(kubectl get all --all-namespaces 2>&1)
    if [ $? -ne 0 ]; then
        echo "Warning: Unable to get Kubernetes resources. This might be due to
 permissions."
```

```
        echo "Error details: $KUBE_RESOURCES"
        echo "Continuing with script execution..."
    else
        echo "$KUBE_RESOURCES"
    fi
fi

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="

# Prompt for cleanup
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    cleanup_resources
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run the following commands:"
    echo "1. Delete node group: aws eks delete-nodegroup --cluster-name
 $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "2. Wait for node group deletion: aws eks wait nodegroup-deleted --
cluster-name $CLUSTER_NAME --nodegroup-name $NODEGROUP_NAME"
    echo "3. Delete cluster: aws eks delete-cluster --name $CLUSTER_NAME"
    echo "4. Wait for cluster deletion: aws eks wait cluster-deleted --name
 $CLUSTER_NAME"
    echo "5. Delete CloudFormation stack: aws cloudformation delete-stack --
stack-name $STACK_NAME"
    echo "6. Detach and delete IAM roles for the node group and cluster"
fi

echo "Script completed at $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateCluster
  - CreateNodegroup
  - CreateRole
  - CreateStack
  - DeleteCluster
  - DeleteNodegroup
  - DeleteRole
  - DeleteStack
  - DescribeCluster
  - DescribeNodegroup
  - DescribeStacks
  - DetachRolePolicy
  - GetCallerIdentity
  - GetRole
  - ListNodegroups
  - UpdateKubeconfig
  - Espere (CloudFormation)
  - Aguardar (Amazon EKS)

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de streaming gerenciado

O código de exemplo abaixo mostra como:

- Criar um cluster do Amazon MSK.
- Criar permissões do IAM para acessar o MSK.

- Criar uma máquina cliente
- Obter os agentes de bootstrap.
- Configurar a máquina cliente.
- Criar um tópico e produzir/consumir dados.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon MSK Getting Started Tutorial Script - Version 8
# This script automates the steps in the Amazon MSK Getting Started tutorial
# It creates an MSK cluster, sets up IAM permissions, creates a client machine,
# and configures the client to interact with the cluster

# Set up logging
LOG_FILE="msk_tutorial_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon MSK Getting Started Tutorial Script - Version 8"
echo "Logging to $LOG_FILE"
echo "================================================"

# Function to handle errors
handle_error() {
    echo "ERROR: $1"
    echo "Resources created so far:"
    if [ -n "$CLUSTER_ARN" ]; then echo "- MSK Cluster: $CLUSTER_ARN"; fi
    if [ -n "$POLICY_ARN" ]; then echo "- IAM Policy: $POLICY_ARN"; fi
    if [ -n "$ROLE_NAME" ]; then echo "- IAM Role: $ROLE_NAME"; fi
    if [ -n "$INSTANCE_PROFILE_NAME" ]; then echo "- IAM Instance Profile:
 $INSTANCE_PROFILE_NAME"; fi
```

```
    if [ -n "$CLIENT_SG_ID" ]; then echo "- Client Security Group:
 $CLIENT_SG_ID"; fi
    if [ -n "$INSTANCE_ID" ]; then echo "- EC2 Instance: $INSTANCE_ID"; fi
    if [ -n "$KEY_NAME" ]; then echo "- Key Pair: $KEY_NAME"; fi

    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to check if a resource exists
resource_exists() {
    local resource_type="$1"
    local resource_id="$2"

    case "$resource_type" in
        "cluster")
            aws kafka describe-cluster --cluster-arn "$resource_id" &>/dev/null
            ;;
        "policy")
            aws iam get-policy --policy-arn "$resource_id" &>/dev/null
            ;;
        "role")
            aws iam get-role --role-name "$resource_id" &>/dev/null
            ;;
        "instance-profile")
            aws iam get-instance-profile --instance-profile-name "$resource_id"
 &>/dev/null
            ;;
        "security-group")
            aws ec2 describe-security-groups --group-ids "$resource_id" &>/dev/
null
            ;;
        "instance")
            aws ec2 describe-instances --instance-ids "$resource_id" --query
 'Reservations[0].Instances[0].State.Name' --output text | grep -v "terminated"
 &>/dev/null
            ;;
        "key-pair")
            aws ec2 describe-key-pairs --key-names "$resource_id" &>/dev/null
            ;;
    esac
}
```

```
# Function to remove security group references
remove_security_group_references() {
    local sg_id="$1"

    if [ -z "$sg_id" ]; then
        echo "No security group ID provided for reference removal"
        return
    fi

    echo "Removing security group references for $sg_id"

    # Get all security groups in the VPC that might reference our client security
 group
    local vpc_security_groups=$(aws ec2 describe-security-groups \
        --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" \
        --query 'SecurityGroups[].GroupId' \
        --output text 2>/dev/null)

    if [ -n "$vpc_security_groups" ]; then
        for other_sg in $vpc_security_groups; do
            if [ "$other_sg" != "$sg_id" ]; then
                echo "Checking security group $other_sg for references to $sg_id"

                # Get the security group details in JSON format
                local sg_details=$(aws ec2 describe-security-groups \
                    --group-ids "$other_sg" \
                    --output json 2>/dev/null)

                if [ -n "$sg_details" ]; then
                    # Check if our security group is referenced in inbound rules
                    local has_inbound_ref=$(echo "$sg_details" | grep -o
 "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_inbound_ref" ]; then
                        echo "Found inbound rules in $other_sg referencing
 $sg_id, removing them..."

                        # Try to remove common rule types
                        echo "Attempting to remove all-traffic rule"
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No all-
traffic rule to remove"
```

```
                        # Try to remove TCP rules on common ports
                        for port in 22 80 443 9092 9094 9096; do
                            aws ec2 revoke-security-group-ingress \
                                --group-id "$other_sg" \
                                --protocol tcp \
                                --port "$port" \
                                --source-group "$sg_id" 2>/dev/null || true
                        done

                        # Try to remove UDP rules
                        aws ec2 revoke-security-group-ingress \
                            --group-id "$other_sg" \
                            --protocol udp \
                            --source-group "$sg_id" 2>/dev/null || true
                    fi

                    # Check for outbound rules (less common but possible)
                    local has_outbound_ref=$(echo "$sg_details" | grep -A 20
 "IpPermissionsEgress" | grep -o "\"GroupId\": \"$sg_id\"" | head -1)

                    if [ -n "$has_outbound_ref" ]; then
                        echo "Found outbound rules in $other_sg referencing
 $sg_id, removing them..."

                        aws ec2 revoke-security-group-egress \
                            --group-id "$other_sg" \
                            --protocol all \
                            --source-group "$sg_id" 2>/dev/null || echo "No
 outbound all-traffic rule to remove"
                    fi
                fi
            fi
        done
    fi

    echo "Completed security group reference removal for $sg_id"
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    # Delete EC2 instance if it exists
```

```
   if [ -n "$INSTANCE_ID" ] && resource_exists "instance" "$INSTANCE_ID"; then
       echo "Terminating EC2 instance: $INSTANCE_ID"
       aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || echo "Failed
to terminate instance"
       echo "Waiting for instance to terminate..."
       aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID" || echo
"Failed to wait for instance termination"
   fi

   # Delete MSK cluster first (to remove dependencies on security group)
   if [ -n "$CLUSTER_ARN" ] && resource_exists "cluster" "$CLUSTER_ARN"; then
       echo "Deleting MSK cluster: $CLUSTER_ARN"
       aws kafka delete-cluster --cluster-arn "$CLUSTER_ARN" || echo "Failed to
delete cluster"

       # Wait a bit for the cluster deletion to start
       echo "Waiting 30 seconds for cluster deletion to begin..."
       sleep 30
   fi

   # Remove security group references before attempting deletion
   if [ -n "$CLIENT_SG_ID" ] && resource_exists "security-group"
"$CLIENT_SG_ID"; then
       remove_security_group_references "$CLIENT_SG_ID"

       echo "Deleting security group: $CLIENT_SG_ID"
       # Try multiple times with longer delays to ensure dependencies are
removed
       for i in {1..10}; do
           if aws ec2 delete-security-group --group-id "$CLIENT_SG_ID"; then
               echo "Security group deleted successfully"
               break
           fi
           echo "Failed to delete security group (attempt $i/10), retrying in 30
seconds..."
           sleep 30
       done
   fi

   # Delete key pair if it exists
   if [ -n "$KEY_NAME" ] && resource_exists "key-pair" "$KEY_NAME"; then
       echo "Deleting key pair: $KEY_NAME"
       aws ec2 delete-key-pair --key-name "$KEY_NAME" || echo "Failed to delete
key pair"
```

```
    fi

    # Remove role from instance profile
    if [ -n "$ROLE_NAME" ] && [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists
"instance-profile" "$INSTANCE_PROFILE_NAME"; then
        echo "Removing role from instance profile"
        aws iam remove-role-from-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" \
            --role-name "$ROLE_NAME" || echo "Failed to remove role from instance
profile"
    fi

    # Delete instance profile
    if [ -n "$INSTANCE_PROFILE_NAME" ] && resource_exists "instance-profile"
"$INSTANCE_PROFILE_NAME"; then
        echo "Deleting instance profile: $INSTANCE_PROFILE_NAME"
        aws iam delete-instance-profile \
            --instance-profile-name "$INSTANCE_PROFILE_NAME" || echo "Failed to
delete instance profile"
    fi

    # Detach policy from role
    if [ -n "$ROLE_NAME" ] && [ -n "$POLICY_ARN" ] && resource_exists "role"
"$ROLE_NAME"; then
        echo "Detaching policy from role"
        aws iam detach-role-policy \
            --role-name "$ROLE_NAME" \
            --policy-arn "$POLICY_ARN" || echo "Failed to detach policy"
    fi

    # Delete role
    if [ -n "$ROLE_NAME" ] && resource_exists "role" "$ROLE_NAME"; then
        echo "Deleting role: $ROLE_NAME"
        aws iam delete-role --role-name "$ROLE_NAME" || echo "Failed to delete
role"
    fi

    # Delete policy
    if [ -n "$POLICY_ARN" ] && resource_exists "policy" "$POLICY_ARN"; then
        echo "Deleting policy: $POLICY_ARN"
        aws iam delete-policy --policy-arn "$POLICY_ARN" || echo "Failed to
delete policy"
    fi
```

```
    echo "Cleanup completed"
}

# Function to find a suitable subnet and instance type combination
find_suitable_subnet_and_instance_type() {
    local vpc_id="$1"
    local -a subnet_array=("${!2}")

    # List of instance types to try, in order of preference
    local instance_types=("t3.micro" "t2.micro" "t3.small" "t2.small")

    echo "Finding suitable subnet and instance type combination..."

    for instance_type in "${instance_types[@]}"; do
        echo "Trying instance type: $instance_type"

        for subnet_id in "${subnet_array[@]}"; do
            # Get the availability zone for this subnet
            local az=$(aws ec2 describe-subnets \
                --subnet-ids "$subnet_id" \
                --query 'Subnets[0].AvailabilityZone' \
                --output text)

            echo "  Checking subnet $subnet_id in AZ $az"

            # Check if this instance type is available in this AZ
            local available=$(aws ec2 describe-instance-type-offerings \
                --location-type availability-zone \
                --filters "Name=location,Values=$az" "Name=instance-type,Values=
$instance_type" \
                --query 'InstanceTypeOfferings[0].InstanceType' \
                --output text 2>/dev/null)

            if [ "$available" = "$instance_type" ]; then
                echo "  # Found suitable combination: $instance_type in $az
 (subnet: $subnet_id)"
                SELECTED_SUBNET_ID="$subnet_id"
                SELECTED_INSTANCE_TYPE="$instance_type"
                return 0
            else
                echo "  # $instance_type not available in $az"
            fi
        done
    done
```

```
    echo "ERROR: Could not find any suitable subnet and instance type
 combination"
    return 1
}

# Generate unique identifiers
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n 1)
CLUSTER_NAME="MSKTutorialCluster-${RANDOM_SUFFIX}"
POLICY_NAME="msk-tutorial-policy-${RANDOM_SUFFIX}"
ROLE_NAME="msk-tutorial-role-${RANDOM_SUFFIX}"
INSTANCE_PROFILE_NAME="msk-tutorial-profile-${RANDOM_SUFFIX}"
SG_NAME="MSKClientSecurityGroup-${RANDOM_SUFFIX}"

echo "Using the following resource names:"
echo "- Cluster Name: $CLUSTER_NAME"
echo "- Policy Name: $POLICY_NAME"
echo "- Role Name: $ROLE_NAME"
echo "- Instance Profile Name: $INSTANCE_PROFILE_NAME"
echo "- Security Group Name: $SG_NAME"
echo "================================================"

# Step 1: Create an MSK Provisioned cluster
echo "Step 1: Creating MSK Provisioned cluster"

# Get the default VPC ID first
echo "Getting default VPC..."
DEFAULT_VPC_ID=$(aws ec2 describe-vpcs \
    --filters "Name=is-default,Values=true" \
    --query "Vpcs[0].VpcId" \
    --output text)

if [ -z "$DEFAULT_VPC_ID" ] || [ "$DEFAULT_VPC_ID" = "None" ]; then
    handle_error "Could not find default VPC. Please ensure you have a default
 VPC in your region."
fi

echo "Default VPC ID: $DEFAULT_VPC_ID"

# Get available subnets in the default VPC
echo "Getting available subnets in the default VPC..."
SUBNETS=$(aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$DEFAULT_VPC_ID" "Name=default-for-
az,Values=true" \
```

```
    --query "Subnets[0:3].SubnetId" \
    --output text)

# Convert space-separated subnet IDs to an array
read -r -a SUBNET_ARRAY <<< "$SUBNETS"

if [ ${#SUBNET_ARRAY[@]} -lt 3 ]; then
    handle_error "Not enough subnets available in the default VPC. Need at least
 3 subnets, found ${#SUBNET_ARRAY[@]}."
fi

# Get default security group for the default VPC
echo "Getting default security group for the default VPC..."
DEFAULT_SG=$(aws ec2 describe-security-groups \
    --filters "Name=group-name,Values=default" "Name=vpc-id,Values=
$DEFAULT_VPC_ID" \
    --query "SecurityGroups[0].GroupId" \
    --output text)

if [ -z "$DEFAULT_SG" ] || [ "$DEFAULT_SG" = "None" ]; then
    handle_error "Could not find default security group for VPC $DEFAULT_VPC_ID"
fi

echo "Creating MSK cluster: $CLUSTER_NAME"
echo "Using VPC: $DEFAULT_VPC_ID"
echo "Using subnets: ${SUBNET_ARRAY[0]} ${SUBNET_ARRAY[1]} ${SUBNET_ARRAY[2]}"
echo "Using security group: $DEFAULT_SG"

# Create the MSK cluster with proper error handling
CLUSTER_RESPONSE=$(aws kafka create-cluster \
    --cluster-name "$CLUSTER_NAME" \
    --broker-node-group-info "{\"InstanceType\": \"kafka.t3.small\",
 \"ClientSubnets\": [\"${SUBNET_ARRAY[0]}\", \"${SUBNET_ARRAY[1]}\",
 \"${SUBNET_ARRAY[2]}\"], \"SecurityGroups\": [\"$DEFAULT_SG\"]}" \
    --kafka-version "3.6.0" \
    --number-of-broker-nodes 3 \
    --encryption-info "{\"EncryptionInTransit\": {\"InCluster\": true,
 \"ClientBroker\": \"TLS\"}}" \
    --tags project=doc-smith,tutorial=amazon-managed-streaming-for-apache-kafka-
gs 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create MSK cluster: $CLUSTER_RESPONSE"
```

```
fi

# Extract the cluster ARN using grep
CLUSTER_ARN=$(echo "$CLUSTER_RESPONSE" | grep -o '"ClusterArn": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLUSTER_ARN" ]; then
    handle_error "Failed to extract cluster ARN from response: $CLUSTER_RESPONSE"
fi

echo "MSK cluster creation initiated. ARN: $CLUSTER_ARN"
echo "Waiting for cluster to become active (this may take 15-20 minutes)..."

# Wait for the cluster to become active
while true; do
    CLUSTER_STATUS=$(aws kafka describe-cluster --cluster-arn "$CLUSTER_ARN" --
query "ClusterInfo.State" --output text 2>/dev/null)

    if [ $? -ne 0 ]; then
        echo "Failed to get cluster status. Retrying in 30 seconds..."
        sleep 30
        continue
    fi

    echo "Current cluster status: $CLUSTER_STATUS"

    if [ "$CLUSTER_STATUS" = "ACTIVE" ]; then
        echo "Cluster is now active!"
        break
    elif [ "$CLUSTER_STATUS" = "FAILED" ]; then
        handle_error "Cluster creation failed"
    fi

    echo "Still waiting for cluster to become active... (checking again in 60
 seconds)"
    sleep 60
done

echo "============================================="
# Step 2: Create an IAM role granting access to create topics on the Amazon MSK
 cluster
echo "Step 2: Creating IAM policy and role"

# Get account ID and region
```

```
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION=$(aws ec2 describe-availability-zones --query
 'AvailabilityZones[0].RegionName' --output text)
fi

if [ -z "$ACCOUNT_ID" ] || [ -z "$REGION" ]; then
    handle_error "Could not determine AWS account ID or region"
fi

echo "Account ID: $ACCOUNT_ID"
echo "Region: $REGION"

# Create IAM policy
echo "Creating IAM policy: $POLICY_NAME"
POLICY_DOCUMENT="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:Connect\",
                \"kafka-cluster:AlterCluster\",
                \"kafka-cluster:DescribeCluster\"
            ],
            \"Resource\": [
                \"$CLUSTER_ARN\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
                \"kafka-cluster:*Topic*\",
                \"kafka-cluster:WriteData\",
                \"kafka-cluster:ReadData\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:topic/$CLUSTER_NAME/*\"
            ]
        },
        {
            \"Effect\": \"Allow\",
            \"Action\": [
```

```
                \"kafka-cluster:AlterGroup\",
                \"kafka-cluster:DescribeGroup\"
            ],
            \"Resource\": [
                \"arn:aws:kafka:$REGION:$ACCOUNT_ID:group/$CLUSTER_NAME/*\"
            ]
        }
    ]
}"

POLICY_RESPONSE=$(aws iam create-policy \
    --policy-name "$POLICY_NAME" \
    --policy-document "$POLICY_DOCUMENT" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create IAM policy: $POLICY_RESPONSE"
fi

# Extract the policy ARN using grep
POLICY_ARN=$(echo "$POLICY_RESPONSE" | grep -o '"Arn": "[^"]*' | cut -d'"' -f4)

if [ -z "$POLICY_ARN" ]; then
    handle_error "Failed to extract policy ARN from response: $POLICY_RESPONSE"
fi

aws iam tag-policy --policy-arn "$POLICY_ARN" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

echo "IAM policy created. ARN: $POLICY_ARN"

# Create IAM role for EC2
echo "Creating IAM role: $ROLE_NAME"
TRUST_POLICY="{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"ec2.amazonaws.com\"},\"Action\":\"sts:AssumeRole
\"}]}"

ROLE_RESPONSE=$(aws iam create-role \
    --role-name "$ROLE_NAME" \
    --assume-role-policy-document "$TRUST_POLICY" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
```

```
    handle_error "Failed to create IAM role: $ROLE_RESPONSE"
fi

echo "IAM role created: $ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs

# Attach policy to role
echo "Attaching policy to role"
ATTACH_RESPONSE=$(aws iam attach-role-policy \
    --role-name "$ROLE_NAME" \
    --policy-arn "$POLICY_ARN" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to attach policy to role: $ATTACH_RESPONSE"
fi

echo "Policy attached to role"

# Create instance profile and add role to it
echo "Creating instance profile: $INSTANCE_PROFILE_NAME"
PROFILE_RESPONSE=$(aws iam create-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create instance profile: $PROFILE_RESPONSE"
fi

echo "Instance profile created"

echo "Adding role to instance profile"
ADD_ROLE_RESPONSE=$(aws iam add-role-to-instance-profile \
    --instance-profile-name "$INSTANCE_PROFILE_NAME" \
    --role-name "$ROLE_NAME" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to add role to instance profile: $ADD_ROLE_RESPONSE"
fi

echo "Role added to instance profile"
```

```
# Wait a moment for IAM propagation
echo "Waiting 10 seconds for IAM propagation..."
sleep 10

echo "============================================="

# Step 3: Create a client machine
echo "Step 3: Creating client machine"

# Find a suitable subnet and instance type combination
if ! find_suitable_subnet_and_instance_type "$DEFAULT_VPC_ID" SUBNET_ARRAY[@];
 then
    handle_error "Could not find a suitable subnet and instance type combination"
fi

echo "Selected subnet: $SELECTED_SUBNET_ID"
echo "Selected instance type: $SELECTED_INSTANCE_TYPE"

# Verify the subnet is in the same VPC we're using
SUBNET_VPC_ID=$(aws ec2 describe-subnets \
    --subnet-ids "$SELECTED_SUBNET_ID" \
    --query 'Subnets[0].VpcId' \
    --output text)

if [ "$SUBNET_VPC_ID" != "$DEFAULT_VPC_ID" ]; then
    handle_error "Subnet VPC ($SUBNET_VPC_ID) does not match default VPC
 ($DEFAULT_VPC_ID)"
fi

echo "VPC ID: $SUBNET_VPC_ID"

# Get security group ID from the MSK cluster
echo "Getting security group ID from the MSK cluster"
MSK_SG_ID=$(aws kafka describe-cluster \
    --cluster-arn "$CLUSTER_ARN" \
    --query 'ClusterInfo.BrokerNodeGroupInfo.SecurityGroups[0]' \
    --output text)

if [ -z "$MSK_SG_ID" ] || [ "$MSK_SG_ID" = "None" ]; then
    handle_error "Failed to get security group ID from cluster"
fi

echo "MSK security group ID: $MSK_SG_ID"
```

```
# Create security group for client
echo "Creating security group for client: $SG_NAME"
CLIENT_SG_RESPONSE=$(aws ec2 create-security-group \
    --group-name "$SG_NAME" \
    --description "Security group for MSK client" \
    --vpc-id "$DEFAULT_VPC_ID" \
    --tag-specifications 'ResourceType=security-
group,Tags=[{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-managed-
streaming-for-apache-kafka-gs}]' 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create security group: $CLIENT_SG_RESPONSE"
fi

# Extract the security group ID using grep
CLIENT_SG_ID=$(echo "$CLIENT_SG_RESPONSE" | grep -o '"GroupId": "[^"]*' | cut -
d'"' -f4)

if [ -z "$CLIENT_SG_ID" ]; then
    handle_error "Failed to extract security group ID from response:
 $CLIENT_SG_RESPONSE"
fi

echo "Client security group created. ID: $CLIENT_SG_ID"

# Allow SSH access to client from your IP only
echo "Getting your public IP address"
MY_IP=$(curl -s https://checkip.amazonaws.com 2>/dev/null)

if [ -z "$MY_IP" ]; then
    echo "Warning: Could not determine your IP address. Using 0.0.0.0/0 (not
 recommended for production)"
    MY_IP="0.0.0.0/0"
else
    MY_IP="$MY_IP/32"
    echo "Your IP address: $MY_IP"
fi

echo "Adding SSH ingress rule to client security group"
SSH_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$CLIENT_SG_ID" \
    --protocol tcp \
```

```
    --port 22 \
    --cidr "$MY_IP" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add SSH ingress rule: $SSH_RULE_RESPONSE"
    echo "You may need to manually add SSH access to security group
 $CLIENT_SG_ID"
fi

echo "SSH ingress rule added"

# Update MSK security group to allow traffic from client security group
echo "Adding ingress rule to MSK security group to allow traffic from client"
MSK_RULE_RESPONSE=$(aws ec2 authorize-security-group-ingress \
    --group-id "$MSK_SG_ID" \
    --protocol all \
    --source-group "$CLIENT_SG_ID" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    echo "Warning: Failed to add ingress rule to MSK security group:
 $MSK_RULE_RESPONSE"
    echo "You may need to manually add ingress rule to security group $MSK_SG_ID"
fi

echo "Ingress rule added to MSK security group"

# Create key pair
KEY_NAME="MSKKeyPair-${RANDOM_SUFFIX}"
echo "Creating key pair: $KEY_NAME"
KEY_RESPONSE=$(aws ec2 create-key-pair --key-name "$KEY_NAME" --tag-
specifications 'ResourceType=key-pair,Tags=[{Key=project,Value=doc-smith},
{Key=tutorial,Value=amazon-managed-streaming-for-apache-kafka-gs}]' --query
 'KeyMaterial' --output text 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to create key pair: $KEY_RESPONSE"
fi

# Save the private key to a file
KEY_FILE="${KEY_NAME}.pem"
echo "$KEY_RESPONSE" > "$KEY_FILE"
```

```
chmod 400 "$KEY_FILE"

echo "Key pair created and saved to $KEY_FILE"

# Get the latest Amazon Linux 2 AMI
echo "Getting latest Amazon Linux 2 AMI ID"
AMI_ID=$(aws ec2 describe-images \
    --owners amazon \
    --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
    --query "sort_by(Images, &CreationDate)[-1].ImageId" \
    --output text 2>/dev/null)

if [ -z "$AMI_ID" ] || [ "$AMI_ID" = "None" ]; then
    handle_error "Failed to get Amazon Linux 2 AMI ID"
fi

echo "Using AMI ID: $AMI_ID"

# Launch EC2 instance with the selected subnet and instance type
echo "Launching EC2 instance"
echo "Instance type: $SELECTED_INSTANCE_TYPE"
echo "Subnet: $SELECTED_SUBNET_ID"

INSTANCE_RESPONSE=$(aws ec2 run-instances \
    --image-id "$AMI_ID" \
    --instance-type "$SELECTED_INSTANCE_TYPE" \
    --key-name "$KEY_NAME" \
    --security-group-ids "$CLIENT_SG_ID" \
    --subnet-id "$SELECTED_SUBNET_ID" \
    --iam-instance-profile "Name=$INSTANCE_PROFILE_NAME" \
    --tag-specifications
 "ResourceType=instance,Tags=[{Key=Name,Value=MSKTutorialClient-
${RANDOM_SUFFIX}},{Key=project,Value=doc-smith},{Key=tutorial,Value=amazon-
managed-streaming-for-apache-kafka-gs}]" 2>&1)

# Check if the command was successful
if [ $? -ne 0 ]; then
    handle_error "Failed to launch EC2 instance: $INSTANCE_RESPONSE"
fi

# Extract the instance ID using grep
INSTANCE_ID=$(echo "$INSTANCE_RESPONSE" | grep -o '"InstanceId": "[^"]*' | head
 -1 | cut -d'"' -f4)
```

```
if [ -z "$INSTANCE_ID" ]; then
    handle_error "Failed to extract instance ID from response:
 $INSTANCE_RESPONSE"
fi

echo "EC2 instance launched successfully. ID: $INSTANCE_ID"
echo "Waiting for instance to be running..."

# Wait for the instance to be running
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"

if [ $? -ne 0 ]; then
    handle_error "Instance failed to reach running state"
fi

# Wait a bit more for the instance to initialize
echo "Instance is running. Waiting 30 seconds for initialization..."
sleep 30

# Get public DNS name of instance
CLIENT_DNS=$(aws ec2 describe-instances \
    --instance-ids "$INSTANCE_ID" \
    --query 'Reservations[0].Instances[0].PublicDnsName' \
    --output text)

if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
    echo "Warning: Could not get public DNS name for instance. Trying public
 IP..."
    CLIENT_DNS=$(aws ec2 describe-instances \
        --instance-ids "$INSTANCE_ID" \
        --query 'Reservations[0].Instances[0].PublicIpAddress' \
        --output text)

    if [ -z "$CLIENT_DNS" ] || [ "$CLIENT_DNS" = "None" ]; then
        handle_error "Failed to get public DNS name or IP address for instance"
    fi
fi

echo "Client instance DNS/IP: $CLIENT_DNS"
echo "============================================="
# Get bootstrap brokers with improved logic
echo "Getting bootstrap brokers"
MAX_RETRIES=10
```

```
RETRY_COUNT=0
BOOTSTRAP_BROKERS=""
AUTH_METHOD=""

while [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; do
    # Get the full bootstrap brokers response
    BOOTSTRAP_RESPONSE=$(aws kafka get-bootstrap-brokers \
        --cluster-arn "$CLUSTER_ARN" 2>/dev/null)

    if [ $? -eq 0 ] && [ -n "$BOOTSTRAP_RESPONSE" ]; then
        # Try to get IAM authentication brokers first using grep
        BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringSaslIam": "[^"]*' | cut -d'"' -f4)
        if [ -n "$BOOTSTRAP_BROKERS" ]; then
            AUTH_METHOD="IAM"
        else
            # Fall back to TLS authentication
            BOOTSTRAP_BROKERS=$(echo "$BOOTSTRAP_RESPONSE" | grep -o
 '"BootstrapBrokerStringTls": "[^"]*' | cut -d'"' -f4)
            if [ -n "$BOOTSTRAP_BROKERS" ]; then
                AUTH_METHOD="TLS"
            fi
        fi
    fi

    RETRY_COUNT=$((RETRY_COUNT + 1))

    if [ "$RETRY_COUNT" -ge "$MAX_RETRIES" ]; then
        echo "Warning: Could not get bootstrap brokers after $MAX_RETRIES
 attempts."
        echo "You may need to manually retrieve them later using:"
        echo "aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
        BOOTSTRAP_BROKERS="BOOTSTRAP_BROKERS_NOT_AVAILABLE"
        AUTH_METHOD="UNKNOWN"
        break
    fi

    if [ -z "$BOOTSTRAP_BROKERS" ] || [ "$BOOTSTRAP_BROKERS" = "None" ]; then
        echo "Bootstrap brokers not available yet. Retrying in 30 seconds...
 (Attempt $RETRY_COUNT/$MAX_RETRIES)"
        sleep 30
    fi
done
```

```
echo "Bootstrap brokers: $BOOTSTRAP_BROKERS"
echo "Authentication method: $AUTH_METHOD"
echo "============================================="

# Create setup script for the client machine
echo "Creating setup script for the client machine"
cat > setup_client.sh << 'EOF'
#!/bin/bash

# Set up logging
LOG_FILE="client_setup_$(date +%Y%m%d_%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting client setup"
echo "============================================="

# Install Java
echo "Installing Java"
sudo yum -y install java-11

# Set environment variables
echo "Setting up environment variables"
export KAFKA_VERSION="3.6.0"
echo "KAFKA_VERSION=$KAFKA_VERSION"

# Download and extract Apache Kafka
echo "Downloading Apache Kafka"
wget https://archive.apache.org/dist/kafka/$KAFKA_VERSION/kafka_2.13-
$KAFKA_VERSION.tgz
if [ $? -ne 0 ]; then
    echo "Failed to download Kafka. Trying alternative mirror..."
    wget https://www.apache.org/dyn/closer.cgi?path=/kafka/$KAFKA_VERSION/
kafka_2.13-$KAFKA_VERSION.tgz
fi

echo "Extracting Kafka"
tar -xzf kafka_2.13-$KAFKA_VERSION.tgz
export KAFKA_ROOT=$(pwd)/kafka_2.13-$KAFKA_VERSION
echo "KAFKA_ROOT=$KAFKA_ROOT"

# Download the MSK IAM authentication package (needed for both IAM and TLS)
echo "Downloading MSK IAM authentication package"
cd $KAFKA_ROOT/libs
```

```
wget https://github.com/aws/aws-msk-iam-auth/releases/latest/download/aws-msk-
iam-auth-1.1.6-all.jar
if [ $? -ne 0 ]; then
    echo "Failed to download specific version. Trying to get latest version..."
    LATEST_VERSION=$(curl -s https://api.github.com/repos/aws/aws-msk-iam-auth/
releases/latest | grep -o '"tag_name": "[^"]*' | cut -d'"' -f4)
    wget https://github.com/aws/aws-msk-iam-auth/releases/download/
$LATEST_VERSION/aws-msk-iam-auth-$LATEST_VERSION-all.jar
    if [ $? -ne 0 ]; then
        echo "Failed to download IAM auth package. Please check the URL and try
 again."
        exit 1
    fi
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-$LATEST_VERSION-all.jar
else
    export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
fi
echo "CLASSPATH=$CLASSPATH"

# Create client properties file based on authentication method
echo "Creating client properties file"
cd $KAFKA_ROOT/config

# The AUTH_METHOD_PLACEHOLDER will be replaced by the script
AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "Configuring for IAM authentication"
    cat > client.properties << 'EOT'
security.protocol=SASL_SSL
sasl.mechanism=AWS_MSK_IAM
sasl.jaas.config=software.amazon.msk.auth.iam.IAMLoginModule required;
sasl.client.callback.handler.class=software.amazon.msk.auth.iam.IAMClientCallbackHandler
EOT
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "Configuring for TLS authentication"
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
else
    echo "Unknown authentication method. Creating basic TLS configuration."
    cat > client.properties << 'EOT'
security.protocol=SSL
EOT
```

```
fi

echo "Client setup completed"
echo "============================================="

# Create a script to set up environment variables
cat > ~/setup_env.sh << 'EOT'
#!/bin/bash
export KAFKA_VERSION="3.6.0"
export KAFKA_ROOT=~/kafka_2.13-$KAFKA_VERSION
export CLASSPATH=$KAFKA_ROOT/libs/aws-msk-iam-auth-1.1.6-all.jar
export BOOTSTRAP_SERVER="BOOTSTRAP_SERVER_PLACEHOLDER"
export AUTH_METHOD="AUTH_METHOD_PLACEHOLDER"

echo "Environment variables set:"
echo "KAFKA_VERSION=$KAFKA_VERSION"
echo "KAFKA_ROOT=$KAFKA_ROOT"
echo "CLASSPATH=$CLASSPATH"
echo "BOOTSTRAP_SERVER=$BOOTSTRAP_SERVER"
echo "AUTH_METHOD=$AUTH_METHOD"
EOT

chmod +x ~/setup_env.sh

echo "Created environment setup script: ~/setup_env.sh"
echo "Run 'source ~/setup_env.sh' to set up your environment"
EOF

# Replace placeholders in the setup script
if [ -n "$BOOTSTRAP_BROKERS" ] && [ "$BOOTSTRAP_BROKERS" != "None" ] &&
 [ "$BOOTSTRAP_BROKERS" != "BOOTSTRAP_BROKERS_NOT_AVAILABLE" ]; then
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|$BOOTSTRAP_BROKERS|g" setup_client.sh
else
    # If bootstrap brokers are not available, provide instructions to get them
    sed -i "s|BOOTSTRAP_SERVER_PLACEHOLDER|\$(aws kafka get-bootstrap-brokers
 --cluster-arn $CLUSTER_ARN --query 'BootstrapBrokerStringTls' --output text)|g"
 setup_client.sh
fi

# Replace auth method placeholder
sed -i "s|AUTH_METHOD_PLACEHOLDER|$AUTH_METHOD|g" setup_client.sh

echo "Setup script created"
echo "============================================="
```

```
# Display summary of created resources
echo ""
echo "================================================"
echo "RESOURCE SUMMARY"
echo "================================================"
echo "MSK Cluster ARN: $CLUSTER_ARN"
echo "MSK Cluster Name: $CLUSTER_NAME"
echo "Authentication Method: $AUTH_METHOD"
echo "IAM Policy ARN: $POLICY_ARN"
echo "IAM Role Name: $ROLE_NAME"
echo "IAM Instance Profile: $INSTANCE_PROFILE_NAME"
echo "Client Security Group: $CLIENT_SG_ID"
echo "EC2 Instance ID: $INSTANCE_ID"
echo "EC2 Instance Type: $SELECTED_INSTANCE_TYPE"
echo "EC2 Instance DNS: $CLIENT_DNS"
echo "Key Pair: $KEY_NAME (saved to $KEY_FILE)"
echo "Bootstrap Brokers: $BOOTSTRAP_BROKERS"
echo "================================================"
echo ""

# Instructions for connecting to the instance and setting up the client
echo "NEXT STEPS:"
echo "1. Connect to your EC2 instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS"
echo ""
echo "2. Upload the setup script to your instance:"
echo "   scp -i $KEY_FILE setup_client.sh ec2-user@$CLIENT_DNS:~/"
echo ""
echo "3. Run the setup script on your instance:"
echo "   ssh -i $KEY_FILE ec2-user@$CLIENT_DNS 'chmod +x ~/setup_client.sh && ~/
setup_client.sh'"
echo ""
echo "4. Source the environment setup script:"
echo "   source ~/setup_env.sh"
echo ""

# Provide different instructions based on authentication method
if [ "$AUTH_METHOD" = "IAM" ]; then
    echo "5. Create a Kafka topic (using IAM authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
```

```
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
elif [ "$AUTH_METHOD" = "TLS" ]; then
    echo "5. Create a Kafka topic (using TLS authentication):"
    echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --create \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --command-config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --replication-factor 3 \\"
    echo "     --partitions 1 \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "6. Start a producer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-producer.sh \\"
    echo "     --broker-list \$BOOTSTRAP_SERVER \\"
    echo "     --producer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic"
    echo ""
    echo "7. Start a consumer:"
    echo "   \$KAFKA_ROOT/bin/kafka-console-consumer.sh \\"
    echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
    echo "     --consumer.config \$KAFKA_ROOT/config/client.properties \\"
    echo "     --topic MSKTutorialTopic \\"
    echo "     --from-beginning"
else
    echo "5. Manually retrieve bootstrap brokers and configure authentication:"
    echo "   aws kafka get-bootstrap-brokers --cluster-arn $CLUSTER_ARN"
fi

echo ""
echo "8. Verify the topic was created:"
echo "   \$KAFKA_ROOT/bin/kafka-topics.sh --list \\"
```

```
echo "     --bootstrap-server \$BOOTSTRAP_SERVER \\"
echo "     --command-config \$KAFKA_ROOT/config/client.properties"
echo "============================================="
echo ""

# Ask if user wants to clean up resources
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ $CLEANUP_CHOICE =~ ^[Yy]$ ]]; then
    cleanup_resources
    echo "All resources have been cleaned up."
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later."
    echo "To clean up resources, run this script again and choose 'y' when
 prompted."
fi

echo "Script completed successfully!"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - AuthorizeSecurityGroupIngress
  - CreateCluster
  - CreateInstanceProfile
  - CreateKeyPair
  - CreatePolicy
  - CreateRole
  - CreateSecurityGroup
  - DeleteCluster
  - DeleteInstanceProfile

- DeleteKeyPair
- DeletePolicy
- DeleteRole
- DeleteSecurityGroup
- DescribeAvailabilityZones
- DescribeCluster
- DescribeImages
- DescribeInstanceTypeOfferings
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DescribeSubnets
- DescribeVpcs
- DetachRolePolicy
- GetBootstrapBrokers
- GetCallerIdentity
- GetInstanceProfile
- GetPolicy
- GetRole
- RemoveRoleFromInstanceProfile
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- TerminateInstances
- Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos das notificações push

O código de exemplo abaixo mostra como:

- Criar uma aplicação.
- Habilitar canais de notificações por push.
- Enviar notificações por push
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS End User Messaging Push Getting Started Script
# This script creates an AWS End User Messaging Push application and demonstrates
# how to enable push notification channels and send a test message.
#
# Prerequisites:
# - AWS CLI installed and configured
# - Appropriate IAM permissions for Pinpoint operations
#
# Usage: ./aws-end-user-messaging-gs.sh [--auto-cleanup]

set -uo pipefail

# Security: Set secure umask for created files
umask 0077

# Set up logging
LOG_DIR="./aws-eump-logs"
mkdir -p "$LOG_DIR"
LOG_FILE="$LOG_DIR/aws-end-user-messaging-push-script-$(date +%Y%m%d-%H%M%S).log"
```

```
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS End User Messaging Push setup script..."
echo "Logging to $LOG_FILE"
echo "Timestamp: $(date)"

# Security: Track created resources for cleanup
declare -a TEMP_FILES=()
declare -a AWS_RESOURCES=()

# Cleanup function with improved security
cleanup() {
    local exit_code=$?
    echo "Cleaning up temporary resources..."

    # Remove temporary files securely
    for temp_file in "${TEMP_FILES[@]+"${TEMP_FILES[@]}"}"; do
        if [ -f "$temp_file" ]; then
            shred -vfz -n 3 "$temp_file" 2>/dev/null || rm -f "$temp_file"
        fi
    done

    # Optionally delete AWS resources
    if [ "${DELETE_AWS_RESOURCES:-false}" = "true" ]; then
        for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
            echo "Deleting AWS resource: $resource"
            aws pinpoint delete-app --application-id "$resource" 2>/dev/null || \
                echo "Warning: Failed to delete application $resource"
        done
    fi

    exit "$exit_code"
}

trap cleanup EXIT INT TERM

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2
    local ignore_error=${3:-false}

    if echo "$output" | grep -qi "error\|exception\|fail"; then
        echo "ERROR: Command failed: $cmd" >&2
```

```
        echo "Error details: $output" >&2

        if [ "$ignore_error" = "true" ]; then
            echo "Ignoring error and continuing..." >&2
            return 1
        else
            return 2
        fi
    fi

    return 0
}

# Function to validate AWS CLI is configured
validate_aws_cli() {
    echo "Validating AWS CLI configuration..."

    # Check if AWS CLI is installed
    if ! command -v aws &> /dev/null; then
        echo "ERROR: AWS CLI is not installed. Please install it first." >&2
        echo "Visit: https://docs.aws.amazon.com/cli/latest/userguide/getting-
started-install.html" >&2
        return 1
    fi

    # Check AWS CLI version
    AWS_VERSION=$(aws --version 2>&1 | head -n1)
    echo "AWS CLI version: $AWS_VERSION"

    # Verify credentials are set (check for credential env vars or config file)
    if ! aws sts get-caller-identity &> /dev/null; then
        echo "ERROR: AWS CLI credentials are not configured or invalid." >&2
        echo "Please configure credentials via environment variables, credential
 file, or 'aws configure'" >&2
        return 1
    fi

    # Get current AWS identity and region
    CALLER_IDENTITY=$(aws sts get-caller-identity)
    CURRENT_REGION=$(aws configure get region 2>/dev/null || echo "us-east-1")
    echo "AWS CLI configured for:"
    echo "$CALLER_IDENTITY"
    echo "Current region: $CURRENT_REGION"
    echo ""
```

```
    return 0
}

# Function to check if jq is available for JSON parsing
check_json_tools() {
    if command -v jq &> /dev/null; then
        USE_JQ=true
        echo "jq is available for JSON parsing"
    else
        USE_JQ=false
        echo "jq is not available, using grep for JSON parsing"
        echo "Consider installing jq for better JSON handling: https://
stedolan.github.io/jq/"
    fi
}

# Function to extract JSON values safely
extract_json_value() {
    local json=$1
    local key=$2

    if [ "$USE_JQ" = "true" ]; then
        echo "$json" | jq -r ".ApplicationResponse.$key // empty" 2>/dev/null ||
 echo ""
    else
        # Fallback to grep method with better validation
        echo "$json" | grep -o "\"$key\": \"[^\"]*" | cut -d'"' -f4 | head -n1 ||
 echo ""
    fi
}

# Function to validate required IAM permissions
validate_permissions() {
    echo "Validating IAM permissions..."

    # Test basic Pinpoint permissions
    if ! aws pinpoint get-apps > /dev/null 2>&1; then
        echo "WARNING: Unable to list Pinpoint applications." >&2
        echo "Please ensure you have appropriate IAM permissions for Pinpoint
 operations." >&2
        echo "Required permissions:" >&2
        echo "  - mobiletargeting:GetApps" >&2
        echo "  - mobiletargeting:CreateApp" >&2
```

```
        echo "  - mobiletargeting:DeleteApp" >&2
        echo "  - mobiletargeting:UpdateGcmChannel" >&2
        echo "  - mobiletargeting:UpdateApnsChannel" >&2
        echo "  - mobiletargeting:SendMessages" >&2
    else
        echo "Basic Pinpoint permissions validated."
    fi
}

# Function to validate input parameters
validate_input() {
    local app_name=$1

    # Validate app name length and characters
    if [ ${#app_name} -gt 64 ]; then
        echo "ERROR: Application name exceeds maximum length of 64 characters"
 >&2
        return 1
    fi

    if ! [[ "$app_name" =~ ^[a-zA-Z0-9_-]+$ ]]; then
        echo "ERROR: Application name contains invalid characters" >&2
        return 1
    fi

    return 0
}

# Function to create secure temporary files
create_temp_file() {
    local temp_file
    temp_file=$(mktemp) || {
        echo "ERROR: Failed to create temporary file" >&2
        return 1
    }
    chmod 600 "$temp_file"
    TEMP_FILES+=("$temp_file")
    echo "$temp_file"
}

# Validate prerequisites
if ! validate_aws_cli; then
    exit 1
fi
```

```
check_json_tools
validate_permissions

# Generate a random suffix for resource names to avoid conflicts
RANDOM_SUFFIX=$(LC_ALL=C tr -dc 'a-z0-9' < /dev/urandom | fold -w 8 | head -n1)
APP_NAME="PushNotificationApp-${RANDOM_SUFFIX}"

# Validate input
if ! validate_input "$APP_NAME"; then
    exit 1
fi

echo "Creating application with name: $APP_NAME"

# Step 1: Create an application
echo "Executing: aws pinpoint create-app --create-application-request Name=
${APP_NAME}"
CREATE_APP_OUTPUT=$(aws pinpoint create-app --create-application-request "Name=
${APP_NAME},tags={project=doc-smith,tutorial=aws-end-user-messaging-gs}" 2>&1)

if ! check_error "$CREATE_APP_OUTPUT" "create-app"; then
    exit 1
fi

echo "Application created successfully:"
echo "$CREATE_APP_OUTPUT"

# Extract the application ID from the output
APP_ID=$(extract_json_value "$CREATE_APP_OUTPUT" "Id")

if [ -z "$APP_ID" ] || [ "$APP_ID" = "null" ]; then
    echo "ERROR: Failed to extract application ID from output" >&2
    echo "Output was: $CREATE_APP_OUTPUT" >&2
    exit 1
fi

echo "Application ID: $APP_ID"
AWS_RESOURCES+=("$APP_ID")

# Step 2: Enable FCM (GCM) channel with a sample API key
echo ""
echo "========================================="
echo "ENABLING FCM (GCM) CHANNEL"
```

```
echo "============================================="
echo "Note: This is using a placeholder API key for demonstration purposes only."
echo "In a production environment, you should use your actual FCM API key from
 Firebase Console."
echo ""
echo "IMPORTANT: The following command will likely fail because we're using a
 placeholder API key."
echo "This is expected behavior for this demonstration script."

echo "Executing: aws pinpoint update-gcm-channel --application-id $APP_ID --gcm-
channel-request ..."
UPDATE_GCM_OUTPUT=$(aws pinpoint update-gcm-channel \
    --application-id "$APP_ID" \
    --gcm-channel-request '{"Enabled": true, "ApiKey": "sample-fcm-api-key-for-
demo-only"}' 2>&1)

# We'll ignore this specific error since we're using a placeholder API key
if check_error "$UPDATE_GCM_OUTPUT" "update-gcm-channel" "true"; then
    echo "FCM channel enabled successfully:"
    echo "$UPDATE_GCM_OUTPUT"
else
    echo "As expected, FCM channel update failed with the placeholder API key."
    echo "Error details: $UPDATE_GCM_OUTPUT"
    echo ""
    echo "To enable FCM in production:"
    echo "1. Go to Firebase Console (https://console.firebase.google.com/)"
    echo "2. Create or select your project"
    echo "3. Go to Project Settings > Cloud Messaging"
    echo "4. Copy the Server Key"
    echo "5. Replace 'sample-fcm-api-key-for-demo-only' with your actual Server
 Key"
fi

# Step 3: Try to enable APNS channel (this will also fail without real
 certificates)
echo ""
echo "============================================="
echo "ENABLING APNS CHANNEL (OPTIONAL)"
echo "============================================="
echo "Attempting to enable APNS channel with placeholder certificate..."
echo "This will also fail without real APNS certificates, which is expected."

# Create a placeholder APNS configuration
```

```
echo "Executing: aws pinpoint update-apns-channel --application-id $APP_ID --
apns-channel-request ..."
UPDATE_APNS_OUTPUT=$(aws pinpoint update-apns-channel \
    --application-id "$APP_ID" \
    --apns-channel-request '{"Enabled": true, "Certificate": "placeholder-
certificate", "PrivateKey": "placeholder-private-key"}' 2>&1)

if check_error "$UPDATE_APNS_OUTPUT" "update-apns-channel" "true"; then
    echo "APNS channel enabled successfully:"
    echo "$UPDATE_APNS_OUTPUT"
else
    echo "As expected, APNS channel update failed with placeholder certificates."
    echo "Error details: $UPDATE_APNS_OUTPUT"
    echo ""
    echo "To enable APNS in production:"
    echo "1. Generate APNS certificates from Apple Developer Console"
    echo "2. Convert certificates to PEM format"
    echo "3. Use the actual certificate and private key in the update-apns-
channel command"
fi

# Step 4: Create message files for different platforms
echo ""
echo "=========================================="
echo "CREATING MESSAGE FILES"
echo "=========================================="

# Create FCM message file securely
GCM_MESSAGE_FILE=$(create_temp_file)
echo "Creating FCM message file..."
cat > "$GCM_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-FCM": {
      "ChannelType": "GCM"
    }
  },
  "MessageConfiguration": {
    "GCMMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an FCM
 notification.",
      "Priority": "normal",
      "SilentPush": false,
```

```
      "Title": "My First FCM Push Notification",
      "TimeToLive": 30,
      "Data": {
        "key1": "value1",
        "key2": "value2"
      }
    }
  }
}
EOF

# Create APNS message file securely
APNS_MESSAGE_FILE=$(create_temp_file)
echo "Creating APNS message file..."
cat > "$APNS_MESSAGE_FILE" << 'EOF'
{
  "Addresses": {
    "SAMPLE-DEVICE-TOKEN-APNS": {
      "ChannelType": "APNS"
    }
  },
  "MessageConfiguration": {
    "APNSMessage": {
      "Action": "OPEN_APP",
      "Body": "Hello from AWS End User Messaging Push! This is an APNS
 notification.",
      "Priority": "normal",
      "SilentPush": false,
      "Title": "My First APNS Push Notification",
      "TimeToLive": 30,
      "Badge": 1,
      "Sound": "default"
    }
  }
}
EOF

echo "Message files created:"
echo "- FCM message file (for FCM/Android)"
echo "- APNS message file (for APNS/iOS)"
echo ""
echo "Note: These messages use placeholder device tokens and will not actually be
 delivered."
```

```
echo "To send real messages, you would need to replace the sample device tokens
 with actual ones."

# Step 5: Demonstrate how to send messages (this will fail with placeholder
 tokens)
echo ""
echo "========================================="
echo "DEMONSTRATING MESSAGE SENDING"
echo "========================================="
echo "Attempting to send FCM message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<gcm-message>"
SEND_FCM_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$GCM_MESSAGE_FILE" 2>&1)

if check_error "$SEND_FCM_OUTPUT" "send-messages (FCM)" "true"; then
    echo "FCM message sent successfully:"
    echo "$SEND_FCM_OUTPUT"
else
    echo "As expected, FCM message sending failed with placeholder token."
    echo "Error details: $SEND_FCM_OUTPUT"
fi

echo ""
echo "Attempting to send APNS message (will fail with placeholder token)..."

echo "Executing: aws pinpoint send-messages --application-id $APP_ID --message-
request file://<apns-message>"
SEND_APNS_OUTPUT=$(aws pinpoint send-messages \
    --application-id "$APP_ID" \
    --message-request "file://$APNS_MESSAGE_FILE" 2>&1)

if check_error "$SEND_APNS_OUTPUT" "send-messages (APNS)" "true"; then
    echo "APNS message sent successfully:"
    echo "$SEND_APNS_OUTPUT"
else
    echo "As expected, APNS message sending failed with placeholder token."
    echo "Error details: $SEND_APNS_OUTPUT"
fi

# Step 6: Show application details
echo ""
```

```
echo "============================================="
echo "APPLICATION DETAILS"
echo "============================================="
echo "Retrieving application details..."

echo "Executing: aws pinpoint get-app --application-id $APP_ID"
GET_APP_OUTPUT=$(aws pinpoint get-app --application-id "$APP_ID" 2>&1)
if check_error "$GET_APP_OUTPUT" "get-app"; then
    echo "Application details:"
    echo "$GET_APP_OUTPUT"
fi

# Display summary of created resources
echo ""
echo "============================================="
echo "RESOURCES CREATED"
echo "============================================="
echo "AWS Resources:"
for resource in "${AWS_RESOURCES[@]+"${AWS_RESOURCES[@]}"}"; do
    echo "- Application: $resource"
done

echo ""
echo "Files created:"
echo "- $LOG_FILE (script log)"

# Auto-cleanup information
echo ""
echo "============================================="
echo "CLEANUP INFORMATION"
echo "============================================="
echo "This script created AWS resources that may incur charges."
echo "AWS resources will be automatically cleaned up on script exit."
echo ""
echo "To manually delete resources later, use:"
echo "  aws pinpoint delete-app --application-id $APP_ID"

# Set flag to delete AWS resources on cleanup
DELETE_AWS_RESOURCES=true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
```

```
echo "This script demonstrated:"
echo "1. Creating an AWS End User Messaging Push application"
echo "2. Attempting to enable FCM and APNS channels (with placeholder
 credentials)"
echo "3. Creating message templates for different platforms"
echo "4. Demonstrating message sending commands (with placeholder tokens)"
echo "5. Retrieving application details"
echo "6. Proper cleanup of resources"
echo ""
echo "Security best practices implemented:"
echo "- Secure temporary file handling with restricted permissions"
echo "- Input validation for application names"
echo "- Error handling with proper exit codes"
echo "- Credential validation before AWS operations"
echo "- Automatic cleanup on script exit"
echo "- Secure file destruction for sensitive data"
echo ""
echo "For production use:"
echo "- Replace placeholder API keys with real FCM server keys"
echo "- Replace placeholder certificates with real APNS certificates"
echo "- Replace placeholder device tokens with real device tokens"
echo "- Use AWS IAM roles instead of long-term credentials"
echo "- Implement comprehensive error handling"
echo "- Store sensitive credentials in AWS Secrets Manager or Parameter Store"
echo "- Enable CloudTrail for audit logging"
echo "- Use VPC endpoints for private AWS API access"
echo ""
echo "Log file: $LOG_FILE"
echo "Script completed at: $(date)"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - CreateApp
  - DeleteApp
  - GetApp
  - GetApps
  - GetCallerIdentity
  - SendMessages
  - UpdateApnsChannel
  - UpdateGcmChannel

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos dos mecanismos de pesquisa e analytics

O código de exemplo abaixo mostra como:

- Criar um domínio do OpenSearch Service.
- Carregar dados em seu domínio.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Amazon OpenSearch Service Getting Started Script - Version 8 Fixed
# This script creates an OpenSearch domain, uploads data, searches documents, and
 cleans up resources
# Based on the tested and working 4-tutorial-final.md

# SECURITY IMPROVEMENTS IN THIS VERSION:
# 1. Removed hardcoded passwords - now generated securely
# 2. Improved access policy to use principle of least privilege
# 3. Added credential masking in logs
# 4. Removed credentials from command output
# 5. Added input validation for AWS region
# 6. Improved error handling and validation
# 7. Added secure temporary file handling
# 8. Removed unnecessary curl insecurity flags
# 9. Added better secret management practices
# 10. Improved resource tagging for better governance
```

```
set -e  # Exit on any error

# Set up logging with restricted permissions
LOG_FILE="opensearch_tutorial_v8_fixed.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting Amazon OpenSearch Service tutorial script v8-fixed at $(date)"
echo "All commands and outputs will be logged to $LOG_FILE"

# Track if domain was successfully created
DOMAIN_CREATED=false
DOMAIN_ACTIVE=false

# Secure temporary directory
TEMP_DIR=$(mktemp -d)
trap 'rm -rf "$TEMP_DIR"' EXIT
chmod 700 "$TEMP_DIR"

# Error handling function
handle_error() {
    echo "ERROR: $1"
    echo "Attempting to clean up resources..."
    cleanup_resources
    exit 1
}

# Function to clean up resources
cleanup_resources() {
    echo "Cleaning up resources..."

    if [[ "$DOMAIN_CREATED" == "true" ]]; then
        echo "Checking if domain $DOMAIN_NAME exists before attempting to
 delete..."

        # Check if domain exists before trying to delete
        if aws opensearch describe-domain --domain-name "$DOMAIN_NAME" > /dev/
null 2>&1; then
            echo "Domain $DOMAIN_NAME exists. Proceeding with deletion."
            aws opensearch delete-domain --domain-name "$DOMAIN_NAME"
            echo "Domain deletion initiated. This may take several minutes to
 complete."
```

```
        else
            echo "Domain $DOMAIN_NAME does not exist or is not accessible. No
 deletion needed."
        fi
    else
        echo "No domain was successfully created. Nothing to clean up."
    fi

    # Securely clean up sensitive files
    if [[ -d "$TEMP_DIR" ]]; then
        shred -vfz -n 3 "$TEMP_DIR"/* 2>/dev/null || true
        rm -rf "$TEMP_DIR"
    fi
}

# Generate a random identifier for resource names to avoid conflicts
RANDOM_ID=$(openssl rand -hex 4)
DOMAIN_NAME="movies-${RANDOM_ID}"
MASTER_USER="master-user"

# Generate secure password using openssl instead of hardcoding
MASTER_PASSWORD=$(openssl rand -base64 32 | tr -d "=+/" | cut -c1-25)

# Mask password in logs
echo "Using domain name: $DOMAIN_NAME"
echo "Using master username: $MASTER_USER"
echo "Using master password: [REDACTED]"

# Get AWS account ID (matches tutorial)
echo "Retrieving AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
if [[ $? -ne 0 ]] || [[ -z "$ACCOUNT_ID" ]]; then
    handle_error "Failed to retrieve AWS account ID. Please check your AWS
 credentials."
fi
echo "AWS Account ID: $ACCOUNT_ID"

# Get current region (matches tutorial)
echo "Retrieving current AWS region..."
AWS_REGION=$(aws configure get region)
if [[ -z "$AWS_REGION" ]]; then
    AWS_REGION="us-east-1"
    echo "No region found in AWS config, defaulting to $AWS_REGION"
else
```

```
    echo "Using AWS region: $AWS_REGION"
fi

# Validate AWS region format
if ! [[ "$AWS_REGION" =~ ^[a-z]{2}-[a-z]+-[0-9]$ ]]; then
    handle_error "Invalid AWS region format: $AWS_REGION"
fi

# Step 1: Create an OpenSearch Service Domain
echo "Creating OpenSearch Service domain..."
echo "This may take 15-30 minutes to complete."

# SECURITY IMPROVED: Use least privilege access policy
# This policy restricts access to specific actions and should be further
 restricted to specific principals in production
# Create the domain using CloudFormation (handles the 15-20 min creation wait)
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
STACK_NAME="tutorial-opensearch-${RANDOM_ID}"

echo "Creating domain $DOMAIN_NAME via CloudFormation..."
echo "This typically takes 15-20 minutes. CloudFormation will wait for
 completion."
aws cloudformation deploy \
  --template-file "$SCRIPT_DIR/cfn-opensearch-domain.yaml" \
  --stack-name "$STACK_NAME" \
  --parameter-overrides \
    DomainName="$DOMAIN_NAME" \
    MasterUserName="$MASTER_USER" \
    MasterUserPassword="$MASTER_PASSWORD" \
  --tags project=doc-smith tutorial=opensearch-service-gs \
  --no-fail-on-empty-changeset 2>&1 || handle_error "CloudFormation stack
 creation failed"

DOMAIN_CREATED=true
echo "Domain created successfully via CloudFormation."

# Get domain endpoint from stack outputs
DOMAIN_ENDPOINT=$(aws cloudformation describe-stacks \
  --stack-name "$STACK_NAME" \
  --query 'Stacks[0].Outputs[?OutputKey==`DomainEndpoint`].OutputValue' \
  --output text)

if [[ -z "$DOMAIN_ENDPOINT" ]] || [[ "$DOMAIN_ENDPOINT" == "None" ]]; then
    handle_error "Failed to get domain endpoint from CloudFormation outputs"
```

```
fi

echo "Domain endpoint: $DOMAIN_ENDPOINT"

# Wait for fine-grained access control to be fully ready
echo "Waiting for fine-grained access control to initialize..."
sleep 120

# Verify variables are set correctly (matches tutorial)
echo "Verifying configuration..."
echo "Domain endpoint: $DOMAIN_ENDPOINT"
echo "Master user: $MASTER_USER"
echo "Password set: $(if [ -n "$MASTER_PASSWORD" ]; then echo "Yes"; else echo
 "No"; fi)"

# Step 2: Upload Data to the Domain
echo "Preparing to upload data to the domain..."

# Create a file for the single document (matches tutorial exactly)
echo "Creating single document JSON file..."
SINGLE_MOVIE_FILE="$TEMP_DIR/single_movie.json"
cat > "$SINGLE_MOVIE_FILE" << 'EOF'
{
  "director": "Burton, Tim",
  "genre": ["Comedy","Sci-Fi"],
  "year": 1996,
  "actor": ["Jack Nicholson","Pierce Brosnan","Sarah Jessica Parker"],
  "title": "Mars Attacks!"
}
EOF
chmod 600 "$SINGLE_MOVIE_FILE"

# Create a file for bulk documents (matches tutorial exactly)
echo "Creating bulk documents JSON file..."
BULK_MOVIES_FILE="$TEMP_DIR/bulk_movies.json"
cat > "$BULK_MOVIES_FILE" << 'EOF'
{ "index" : { "_index": "movies", "_id" : "2" } }
{"director": "Frankenheimer, John", "genre": ["Drama", "Mystery", "Thriller",
 "Crime"], "year": 1962, "actor": ["Lansbury, Angela", "Sinatra, Frank", "Leigh,
 Janet", "Harvey, Laurence", "Silva, Henry", "Frees, Paul", "Gregory, James",
 "Bissell, Whit", "McGiver, John", "Parrish, Leslie", "Edwards, James", "Flowers,
 Bess", "Dhiegh, Khigh", "Payne, Julie", "Kleeb, Helen", "Gray, Joe", "Nalder,
 Reggie", "Stevens, Bert", "Masters, Michael", "Lowell, Tom"], "title": "The
 Manchurian Candidate"}
```

```
{ "index" : { "_index": "movies", "_id" : "3" } }
{"director": "Baird, Stuart", "genre": ["Action", "Crime", "Thriller"], "year":
 1998, "actor": ["Downey Jr., Robert", "Jones, Tommy Lee", "Snipes, Wesley",
 "Pantoliano, Joe", "Jacob, Irène", "Nelligan, Kate", "Roebuck, Daniel",
 "Malahide, Patrick", "Richardson, LaTanya", "Wood, Tom", "Kosik, Thomas",
 "Stellate, Nick", "Minkoff, Robert", "Brown, Spitfire", "Foster, Reese",
 "Spielbauer, Bruce", "Mukherji, Kevin", "Cray, Ed", "Fordham, David", "Jett,
 Charlie"], "title": "U.S. Marshals"}
{ "index" : { "_index": "movies", "_id" : "4" } }
{"director": "Ray, Nicholas", "genre": ["Drama", "Romance"], "year": 1955,
 "actor": ["Hopper, Dennis", "Wood, Natalie", "Dean, James", "Mineo, Sal",
 "Backus, Jim", "Platt, Edward", "Ray, Nicholas", "Hopper, William", "Allen,
 Corey", "Birch, Paul", "Hudson, Rochelle", "Doran, Ann", "Hicks, Chuck", "Leigh,
 Nelson", "Williams, Robert", "Wessel, Dick", "Bryar, Paul", "Sessions, Almira",
 "McMahon, David", "Peters Jr., House"], "title": "Rebel Without a Cause"}
EOF
chmod 600 "$BULK_MOVIES_FILE"

# Check if curl is installed
if ! command -v curl &> /dev/null; then
    echo "Warning: curl is not installed. Skipping data upload and search steps."
    echo "You can manually upload the data later using the commands in the
 tutorial."
else
    # IMPROVED: Test authentication with multiple approaches
    echo "Testing authentication with the OpenSearch domain..."
    echo "This test checks if fine-grained access control is ready for data
 operations."

    # Create credentials file for secure handling
    CREDENTIALS_FILE="$TEMP_DIR/.credentials"
    echo "$MASTER_USER:$MASTER_PASSWORD" > "$CREDENTIALS_FILE"
    chmod 600 "$CREDENTIALS_FILE"

    # Test 1: Basic authentication with root endpoint
    echo "Testing basic authentication with root endpoint..."
    AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
        --netrc-file "$CREDENTIALS_FILE" \
        --request GET \
        "https://${DOMAIN_ENDPOINT}/" 2>&1)

    echo "Basic auth test result:"
    echo "$AUTH_TEST_RESULT"
```

```
    # Extract HTTP status code
    HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)

    # Function to check if HTTP code is 2xx
    is_success_code() {
        local code=$1
        if [[ "$code" =~ ^2[0-9][0-9]$ ]]; then
            return 0
        else
            return 1
        fi
    }

    # Check if basic authentication test was successful (200 or 2xx status codes)
    if is_success_code "$HTTP_CODE"; then
        echo "# Basic authentication test successful! (HTTP $HTTP_CODE)"
        AUTH_SUCCESS=true
        AUTH_METHOD="basic"
    else
        echo "Basic authentication failed with HTTP code: $HTTP_CODE"

        # Test 2: Try cluster health endpoint
        echo "Testing with cluster health endpoint..."
        HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

        echo "Cluster health test result:"
        echo "$HEALTH_TEST_RESULT"

        HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

        if is_success_code "$HEALTH_HTTP_CODE"; then
            echo "# Cluster health authentication test successful! (HTTP
 $HEALTH_HTTP_CODE)"
            AUTH_SUCCESS=true
            AUTH_METHOD="basic"
        else
            echo "Cluster health authentication also failed with HTTP code:
 $HEALTH_HTTP_CODE"

            # Check for specific error patterns
```

```
            if echo "$AUTH_TEST_RESULT" | grep -q "anonymous is not authorized";
 then
                echo "Error: Request is being treated as anonymous
 (authentication not working)"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Unauthorized"; then
                echo "Error: Authentication credentials rejected"
            elif echo "$AUTH_TEST_RESULT" | grep -q "Forbidden"; then
                echo "Error: Authentication succeeded but access is forbidden"
            fi

            echo "Waiting additional time and retrying with exponential
 backoff..."

            # Retry authentication test with exponential backoff
            AUTH_RETRY_COUNT=0
            MAX_AUTH_RETRIES=5
            WAIT_TIME=60
            AUTH_SUCCESS=false

            while [[ $AUTH_RETRY_COUNT -lt $MAX_AUTH_RETRIES ]]; do
                echo "Retrying authentication test (attempt $((AUTH_RETRY_COUNT
+1))/$MAX_AUTH_RETRIES) after ${WAIT_TIME} seconds..."
                sleep $WAIT_TIME

                # Try both endpoints
                AUTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/" 2>&1)

                HTTP_CODE=$(echo "$AUTH_TEST_RESULT" | grep "HTTP_CODE:" | cut -
d: -f2)

                echo "Retry result (HTTP $HTTP_CODE):"
                echo "$AUTH_TEST_RESULT"

                if is_success_code "$HTTP_CODE"; then
                    echo "# Authentication test successful after retry! (HTTP
 $HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi
```

```
                # Also try cluster health
                HEALTH_TEST_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
                    --netrc-file "$CREDENTIALS_FILE" \
                    --request GET \
                    "https://${DOMAIN_ENDPOINT}/_cluster/health" 2>&1)

                HEALTH_HTTP_CODE=$(echo "$HEALTH_TEST_RESULT" | grep "HTTP_CODE:"
| cut -d: -f2)

                if is_success_code "$HEALTH_HTTP_CODE"; then
                    echo "# Cluster health authentication successful after retry!
(HTTP $HEALTH_HTTP_CODE)"
                    AUTH_SUCCESS=true
                    AUTH_METHOD="basic"
                    break
                fi

                AUTH_RETRY_COUNT=$((AUTH_RETRY_COUNT+1))
                # Exponential backoff: double the wait time each retry (max 10
minutes)
                WAIT_TIME=$((WAIT_TIME * 2))
                if [[ $WAIT_TIME -gt 600 ]]; then
                    WAIT_TIME=600
                fi
            done
        fi
    fi

    # Proceed with data operations if authentication is working
    if [[ "$AUTH_SUCCESS" == "true" ]]; then
        echo "Authentication successful using $AUTH_METHOD method. Proceeding
with data operations."

        # Upload single document (matches tutorial exactly)
        echo "Uploading single document..."
        UPLOAD_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request PUT \
            --header 'Content-Type: application/json' \
            --data @"$SINGLE_MOVIE_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_doc/1" 2>&1)

        echo "Upload response:"
        echo "$UPLOAD_RESULT"
```

```
        UPLOAD_HTTP_CODE=$(echo "$UPLOAD_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        if is_success_code "$UPLOAD_HTTP_CODE" && echo "$UPLOAD_RESULT" | grep -q
 '"result"'; then
            echo "# Single document uploaded successfully! (HTTP
 $UPLOAD_HTTP_CODE)"
        else
            echo "# Warning: Single document upload may have failed (HTTP
 $UPLOAD_HTTP_CODE)"
        fi

        # Upload bulk documents (matches tutorial exactly)
        echo "Uploading bulk documents..."
        BULK_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request POST \
            --header 'Content-Type: application/x-ndjson' \
            --data-binary @"$BULK_MOVIES_FILE" \
            "https://${DOMAIN_ENDPOINT}/movies/_bulk" 2>&1)

        echo "Bulk upload response:"
        echo "$BULK_RESULT"

        BULK_HTTP_CODE=$(echo "$BULK_RESULT" | grep "HTTP_CODE:" | cut -d: -f2)
        if is_success_code "$BULK_HTTP_CODE" && echo "$BULK_RESULT" | grep -q
 '"errors": false'; then
            echo "# Bulk documents uploaded successfully! (HTTP $BULK_HTTP_CODE)"
        else
            echo "# Warning: Bulk document upload may have failed (HTTP
 $BULK_HTTP_CODE)"
        fi

        # Wait a moment for indexing
        echo "Waiting for documents to be indexed..."
        sleep 5

        # Step 3: Search Documents (matches tutorial exactly)
        echo "Searching for documents containing 'mars'..."
        SEARCH_RESULT=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true" 2>&1)
```

```
        SEARCH_HTTP_CODE=$(echo "$SEARCH_RESULT" | grep "HTTP_CODE:" | cut -d: -
f2)
        echo "Search results for 'mars' (HTTP $SEARCH_HTTP_CODE):"
        echo "$SEARCH_RESULT"

        echo "Searching for documents containing 'rebel'..."
        REBEL_SEARCH=$(curl -s -w "\nHTTP_CODE:%{http_code}" \
            --netrc-file "$CREDENTIALS_FILE" \
            --request GET \
            "https://${DOMAIN_ENDPOINT}/movies/_search?q=rebel&pretty=true" 2>&1)

        REBEL_HTTP_CODE=$(echo "$REBEL_SEARCH" | grep "HTTP_CODE:" | cut -d: -f2)
        echo "Search results for 'rebel' (HTTP $REBEL_HTTP_CODE):"
        echo "$REBEL_SEARCH"

        # Verify search results
        if is_success_code "$SEARCH_HTTP_CODE" && echo "$SEARCH_RESULT" | grep -q
 '"hits"'; then
            echo "# Search functionality is working!"
        else
            echo "# Warning: Search may not be working properly."
        fi

    else
        echo ""
        echo "=========================================="
        echo "AUTHENTICATION TROUBLESHOOTING"
        echo "=========================================="
        echo "Authentication failed after all retries. This may be due to:"
        echo "1. Fine-grained access control not fully initialized (most common)"
        echo "2. Domain configuration issues"
        echo "3. Network connectivity issues"
        echo "4. AWS credentials or permissions issues"
        echo ""
        echo "DOMAIN CONFIGURATION DEBUG:"
        echo "Let's check the domain configuration..."

        # Debug domain configuration
        DOMAIN_CONFIG=$(aws opensearch describe-domain --domain-
name "$DOMAIN_NAME" --query 'DomainStatus.{AdvancedSecurityOptions:
 AdvancedSecurityOptions, AccessPolicies: AccessPolicies}' --output json 2>&1)
        echo "Domain configuration:"
        echo "$DOMAIN_CONFIG"
```

```
        echo ""
        echo "MANUAL TESTING COMMANDS:"
        echo "You can try these commands manually in 10-15 minutes:"
        echo ""
        echo "# Test basic authentication:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/\""
        echo ""
        echo "# Test cluster health:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/_cluster/health\""
        echo ""
        echo "# Upload single document:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" --request PUT --
header 'Content-Type: application/json' --data @single_movie.json \"https://
${DOMAIN_ENDPOINT}/movies/_doc/1\""
        echo ""
        echo "# Search for documents:"
        echo "curl --user \"${MASTER_USER}:[PASSWORD]\" \"https://
${DOMAIN_ENDPOINT}/movies/_search?q=mars&pretty=true\""
        echo ""
        echo "TROUBLESHOOTING TIPS:"
        echo "- Wait 10-15 more minutes and try the manual commands"
        echo "- Check AWS CloudTrail logs for authentication errors"
        echo "- Verify your AWS region is correct: $AWS_REGION"
        echo "- Ensure your AWS credentials have OpenSearch permissions"
        echo "- Try accessing OpenSearch Dashboards to verify the master user
 works"
        echo ""
        echo "Skipping data upload and search operations for now."
        echo "The domain is created and accessible via OpenSearch Dashboards."
    fi

    # Securely remove credentials file
    shred -vfz -n 3 "$CREDENTIALS_FILE" 2>/dev/null || true
fi

# Display OpenSearch Dashboards URL (matches tutorial)
echo ""
echo "============================================"
echo "OPENSEARCH DASHBOARDS ACCESS"
echo "============================================"
echo "OpenSearch Dashboards URL: https://${DOMAIN_ENDPOINT}/_dashboards/"
echo "Username: $MASTER_USER"
```

```
echo "Password: [REDACTED - stored securely]"
echo ""
echo "You can access OpenSearch Dashboards using the credentials provided during
 domain creation."
echo "If you uploaded data successfully, you can create an index pattern for
 'movies'."
echo ""

# Summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
echo "OpenSearch Domain Name: $DOMAIN_NAME"
echo "OpenSearch Domain Endpoint: $DOMAIN_ENDPOINT"
echo "AWS Region: $AWS_REGION"
echo "Master Username: $MASTER_USER"
echo ""
echo "ESTIMATED COST: ~$0.038/hour (~$0.91/day) until deleted"
echo ""
echo "Make sure to save the domain name and endpoint for future reference."
echo ""

# Auto-confirm cleanup: Yes, clean up all resources
echo ""
echo "==========================================="
echo "CLEANUP CONFIRMATION"
echo "==========================================="
echo "Automatically cleaning up all created resources..."
CLEANUP_CHOICE="y"

if [[ "${CLEANUP_CHOICE,,}" == "y" ]]; then
    echo "Cleaning up resources..."
    if [ -n "${STACK_NAME:-}" ]; then
        echo "Deleting CloudFormation stack $STACK_NAME..."
        aws cloudformation delete-stack --stack-name "$STACK_NAME"
        echo "# Stack deletion initiated. This may take several minutes."
        echo "  Monitor: aws cloudformation describe-stacks --stack-name
 $STACK_NAME"
    else
        aws opensearch delete-domain --domain-name "$DOMAIN_NAME" 2>/dev/null
        echo "# Domain deletion initiated."
    fi
else
```

```
    echo "Resources will NOT be deleted automatically."
    echo ""
    echo "To delete later:"
    echo "  aws cloudformation delete-stack --stack-name ${STACK_NAME:-tutorial-
opensearch-XXXX}"
    echo ""
    echo "# IMPORTANT: Keeping these resources will incur ongoing AWS charges!"
fi

# Clean up temporary files (handled by trap)
echo "Cleaning up temporary files..."

# Disable the trap since we're handling cleanup manually
trap - EXIT

# Final cleanup
rm -rf "$TEMP_DIR" 2>/dev/null || true

echo ""
echo "============================================="
echo "SCRIPT COMPLETED SUCCESSFULLY"
echo "============================================="
echo "Script completed at $(date)"
echo "All output has been logged to: $LOG_FILE"
echo ""
echo "Next steps:"
echo "1. Access OpenSearch Dashboards at: https://${DOMAIN_ENDPOINT}/
_dashboards/"
echo "2. Create visualizations and dashboards"
echo "3. Explore the OpenSearch API"
echo "4. Remember to delete resources when done to avoid charges"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - CreateDomain
  - DeleteDomain
  - DescribeDomain
  - GetCallerIdentity

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos de firewalls do aplicativo Web

O código de exemplo abaixo mostra como:

- Criar uma ACL da web.
- Adicionar uma regra de correspondência de string.
- Adicionar regras gerenciadas.
- Configurar registro em log da
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS WAF Getting Started Script
# This script creates a Web ACL with a string match rule and AWS Managed Rules,
# associates it with a CloudFront distribution, and then cleans up all resources.

set -euo pipefail

# Security: Restrict file permissions
umask 077

# Set up logging with secure file handling
LOG_FILE="waf-tutorial.log"
touch "$LOG_FILE"
chmod 600 "$LOG_FILE"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "================================================="
echo "AWS WAF Getting Started Tutorial"
echo "================================================="
```

```
echo "This script will create AWS WAF resources and associate"
echo "them with a CloudFront distribution."
echo ""

# Maximum number of retries for operations
MAX_RETRIES=3

# Function to handle errors securely
handle_error() {
    local error_msg="$1"
    echo "ERROR: $error_msg" >&2
    echo "Check the log file for details: $LOG_FILE" >&2
    cleanup_resources
    exit 1
}

# Function to validate AWS CLI JSON output
validate_json() {
    local json_string="$1"
    if ! echo "$json_string" | jq empty 2>/dev/null; then
        return 1
    fi
    return 0
}

# Function to safely extract JSON values
extract_json_value() {
    local json_string="$1"
    local key_path="$2"

    if ! validate_json "$json_string"; then
        return 1
    fi

    echo "$json_string" | jq -r "$key_path" 2>/dev/null || return 1
}

# Function to clean up resources securely
cleanup_resources() {
    echo ""
    echo "=================================================="
    echo "CLEANING UP RESOURCES"
    echo "=================================================="
```

```
    if [ -n "${DISTRIBUTION_ID:-}" ] && [ -n "${WEB_ACL_ARN:-}" ]; then
        echo "Disassociating Web ACL from CloudFront distribution..."
        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID"
            return
        fi

        local disassociate_result
        disassociate_result=$(aws wafv2 disassociate-web-acl \
            --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
            --region us-east-1 2>&1) || true

        if echo "$disassociate_result" | grep -qi "error"; then
            echo "Warning: Failed to disassociate Web ACL: $disassociate_result"
        else
            echo "Web ACL disassociated successfully."
        fi
    fi

    if [ -n "${WEB_ACL_ID:-}" ] && [ -n "${WEB_ACL_NAME:-}" ]; then
        echo "Deleting Web ACL..."

        local get_result
        get_result=$(aws wafv2 get-web-acl \
            --name "$WEB_ACL_NAME" \
            --scope CLOUDFRONT \
            --id "$WEB_ACL_ID" \
            --region us-east-1 2>&1) || true

        if echo "$get_result" | grep -qi "error"; then
            echo "Warning: Failed to get Web ACL for deletion: $get_result"
            echo "You may need to manually delete the Web ACL using the AWS
 Console."
        else
            local latest_token
            latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken'
 2>/dev/null) || latest_token=""

            if [ -n "$latest_token" ]; then
```

```
                local delete_result
                delete_result=$(aws wafv2 delete-web-acl \
                    --name "$WEB_ACL_NAME" \
                    --scope CLOUDFRONT \
                    --id "$WEB_ACL_ID" \
                    --lock-token "$latest_token" \
                    --region us-east-1 2>&1) || true

                if echo "$delete_result" | grep -qi "error"; then
                    echo "Warning: Failed to delete Web ACL: $delete_result"
                    echo "You may need to manually delete the Web ACL using the
 AWS Console."
                else
                    echo "Web ACL deleted successfully."
                fi
            else
                echo "Warning: Could not extract lock token for deletion. You may
 need to manually delete the Web ACL."
            fi
        fi
    fi

    echo "Cleanup process completed."
}

# Security: Trap EXIT to ensure cleanup on any exit
trap cleanup_resources EXIT

# Generate a random identifier for resource names using secure method
RANDOM_ID=$(openssl rand -hex 4) || handle_error "Failed to generate random ID"
WEB_ACL_NAME="MyWebACL-${RANDOM_ID}"
METRIC_NAME="MyWebACLMetrics-${RANDOM_ID}"

echo "Using Web ACL name: $WEB_ACL_NAME"

# Step 1: Create a Web ACL
echo ""
echo "=================================================="
echo "STEP 1: Creating Web ACL"
echo "=================================================="

local create_result
create_result=$(aws wafv2 create-web-acl \
    --name "$WEB_ACL_NAME" \
```

```
    --scope "CLOUDFRONT" \
    --default-action Allow={} \
    --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-waf-gs \
    --region us-east-1 2>&1) || handle_error "Failed to create Web ACL"

if ! validate_json "$create_result"; then
    handle_error "Invalid JSON response from create-web-acl"
fi

# Extract Web ACL ID, ARN, and Lock Token from the response
WEB_ACL_ID=$(extract_json_value "$create_result" '.Summary.Id') || handle_error
 "Failed to extract Web ACL ID"
WEB_ACL_ARN=$(extract_json_value "$create_result" '.Summary.ARN') || handle_error
 "Failed to extract Web ACL ARN"
LOCK_TOKEN=$(extract_json_value "$create_result" '.Summary.LockToken') ||
 handle_error "Failed to extract Lock Token"

echo "Web ACL created successfully with ID: $WEB_ACL_ID"
echo "Lock Token: [REDACTED]"

# Step 2: Add a String Match Rule
echo ""
echo "================================================="
echo "STEP 2: Adding String Match Rule"
echo "================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add string match rule..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
```

```
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
```

```
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add string match rule after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add string match rule: $update_result"
    else
        echo "String match rule added successfully."
        break
    fi
done

# Step 3: Add AWS Managed Rules
echo ""
```

```
echo "================================================="
echo "STEP 3: Adding AWS Managed Rules"
echo "================================================="

for ((i=1; i<=MAX_RETRIES; i++)); do
    echo "Attempt $i to add AWS Managed Rules..."

    # Get the latest lock token before updating
    local get_result
    get_result=$(aws wafv2 get-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope CLOUDFRONT \
        --id "$WEB_ACL_ID" \
        --region us-east-1 2>&1) || true

    if echo "$get_result" | grep -qi "error"; then
        echo "Warning: Failed to get Web ACL for update: $get_result"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to get Web ACL after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    if ! validate_json "$get_result"; then
        echo "Warning: Invalid JSON response from get-web-acl"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Invalid JSON response after $MAX_RETRIES attempts"
        fi
        sleep 2
        continue
    fi

    local latest_token
    latest_token=$(extract_json_value "$get_result" '.WebACL.LockToken' 2>/dev/
null) || true

    if [ -z "$latest_token" ]; then
        echo "Warning: Could not extract lock token for update"
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to extract lock token after $MAX_RETRIES
 attempts"
        fi
        sleep 2
```

```
        continue
    fi

    local update_result
    update_result=$(aws wafv2 update-web-acl \
        --name "$WEB_ACL_NAME" \
        --scope "CLOUDFRONT" \
        --id "$WEB_ACL_ID" \
        --lock-token "$latest_token" \
        --default-action Allow={} \
        --rules '[{
            "Name": "UserAgentRule",
            "Priority": 0,
            "Statement": {
                "ByteMatchStatement": {
                    "SearchString": "MyAgent",
                    "FieldToMatch": {
                        "SingleHeader": {
                            "Name": "user-agent"
                        }
                    },
                    "TextTransformations": [
                        {
                            "Priority": 0,
                            "Type": "NONE"
                        }
                    ],
                    "PositionalConstraint": "EXACTLY"
                }
            },
            "Action": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "UserAgentRuleMetric"
            }
        },
        {
            "Name": "AWS-AWSManagedRulesCommonRuleSet",
            "Priority": 1,
            "Statement": {
                "ManagedRuleGroupStatement": {
```

```
                    "VendorName": "AWS",
                    "Name": "AWSManagedRulesCommonRuleSet",
                    "ExcludedRules": []
                }
            },
            "OverrideAction": {
                "Count": {}
            },
            "VisibilityConfig": {
                "SampledRequestsEnabled": true,
                "CloudWatchMetricsEnabled": true,
                "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
            }
        }]' \
        --visibility-config
 "SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=
$METRIC_NAME" \
        --region us-east-1 2>&1) || true

    if echo "$update_result" | grep -qi "WAFOptimisticLockException"; then
        echo "Optimistic lock exception encountered. Will retry with new lock
 token."
        if [ "$i" -eq "$MAX_RETRIES" ]; then
            handle_error "Failed to add AWS Managed Rules after $MAX_RETRIES
 attempts"
        fi
        sleep 2
        continue
    elif echo "$update_result" | grep -qi "error"; then
        handle_error "Failed to add AWS Managed Rules: $update_result"
    else
        echo "AWS Managed Rules added successfully."
        break
    fi
done

# Step 4: List CloudFront distributions
echo ""
echo "=================================================="
echo "STEP 4: Listing CloudFront Distributions"
echo "=================================================="

local cf_result
```

```
cf_result=$(aws cloudfront list-distributions --query "DistributionList.Items[*].
{Id:Id,DomainName:DomainName}" --output table 2>&1) || cf_result=""

if echo "$cf_result" | grep -qi "error"; then
    echo "Warning: Failed to list CloudFront distributions: $cf_result"
    echo "Continuing without CloudFront association."
    DISTRIBUTION_ID=""
else
    echo "$cf_result"

    # Auto-select first CloudFront distribution if available
    echo ""
    echo "=================================================="
    echo "STEP 5: Associate Web ACL with CloudFront Distribution"
    echo "=================================================="

    local first_dist
    first_dist=$(aws cloudfront list-distributions --query
 "DistributionList.Items[0].Id" --output text 2>&1) || first_dist=""

    if [ -n "$first_dist" ] && [ "$first_dist" != "None" ] && ! echo
 "$first_dist" | grep -qi "error"; then
        DISTRIBUTION_ID="$first_dist"
        echo "Using CloudFront distribution: $DISTRIBUTION_ID"

        local account_id
        account_id=$(aws sts get-caller-identity --query Account --output text
 2>/dev/null) || account_id=""

        if [ -z "$account_id" ]; then
            echo "Warning: Could not retrieve AWS account ID for association"
            DISTRIBUTION_ID=""
        else
            local associate_result
            associate_result=$(aws wafv2 associate-web-acl \
                --web-acl-arn "$WEB_ACL_ARN" \
                --resource-arn "arn:aws:cloudfront::${account_id}:distribution/
${DISTRIBUTION_ID}" \
                --region us-east-1 2>&1) || true

            if echo "$associate_result" | grep -qi "error"; then
                echo "Warning: Failed to associate Web ACL with CloudFront
 distribution: $associate_result"
                echo "Continuing without CloudFront association."
```

```
                DISTRIBUTION_ID=""
            else
                echo "Web ACL associated with CloudFront distribution
 successfully."
            fi
        fi
    else
        echo "No CloudFront distributions available. Skipping association."
        DISTRIBUTION_ID=""
    fi
fi

# Display summary of created resources
echo ""
echo "=============================================="
echo "RESOURCE SUMMARY"
echo "=============================================="
echo "Web ACL Name: $WEB_ACL_NAME"
echo "Web ACL ID: $WEB_ACL_ID"
echo "Web ACL ARN: $WEB_ACL_ARN"
if [ -n "${DISTRIBUTION_ID:-}" ]; then
    echo "Associated CloudFront Distribution: $DISTRIBUTION_ID"
fi
echo ""

# Auto-confirm cleanup
echo "=============================================="
echo "CLEANUP CONFIRMATION"
echo "=============================================="
echo "Proceeding with automatic cleanup of all created resources..."

echo ""
echo "=============================================="
echo "Tutorial completed!"
echo "=============================================="
echo "Log file: $LOG_FILE"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AssociateWebAcl
  - CreateWebAcl
  - DeleteWebAcl
  - DisassociateWebAcl

- GetCallerIdentity
- GetWebAcl
- ListDistributions
- UpdateWebAcl

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como mover segredos codificados para o armazenamento seguro de segredos

O código de exemplo abaixo mostra como:

- Criar perfis do IAM.
- Criar um segredo no Secrets Manager.
- Atualizar o código de uma aplicação.
- Atualizar o segredo.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# Script to move hardcoded secrets to AWS Secrets Manager
# This script demonstrates how to create IAM roles, store a secret in AWS Secrets
 Manager,
# and set up appropriate permissions

set -euo pipefail
```

```
# Set up logging
LOG_FILE="secrets_manager_tutorial.log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS Secrets Manager tutorial script at $(date)"
echo "============================================================"

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -qi "error"; then
        echo "ERROR: Command failed: $cmd"
        echo "$output"
        cleanup_resources
        exit 1
    fi
}

# Function to generate a random identifier using secure method
generate_random_id() {
    python3 -c "import secrets; print('sm' + secrets.token_hex(4))"
}

# Function to safely clean up resources
cleanup_resources() {
    echo ""
    echo "########################################"
    echo "RESOURCES CREATED"
    echo "########################################"

    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Secret: $SECRET_NAME"
    fi

    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "IAM Role: $RUNTIME_ROLE_NAME"
    fi

    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "IAM Role: $ADMIN_ROLE_NAME"
    fi
```

```
    echo ""
    echo "========================================="
    echo "CLEANUP CONFIRMATION"
    echo "========================================="
    echo "Cleaning up all created resources..."

    # Delete secret if it exists
    if [ -n "${SECRET_NAME:-}" ]; then
        echo "Deleting secret: $SECRET_NAME"
        aws secretsmanager delete-secret --secret-id "$SECRET_NAME" --force-
delete-without-recovery 2>/dev/null || true
    fi

    # Detach policies and delete runtime role if it exists
    if [ -n "${RUNTIME_ROLE_NAME:-}" ]; then
        echo "Deleting inline policies from runtime role: $RUNTIME_ROLE_NAME"
        for policy in $(aws iam list-role-policies --role-name
 "$RUNTIME_ROLE_NAME" --query 'PolicyNames[]' --output text 2>/dev/null || true);
 do
            aws iam delete-role-policy --role-name "$RUNTIME_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done
        echo "Deleting IAM role: $RUNTIME_ROLE_NAME"
        aws iam delete-role --role-name "$RUNTIME_ROLE_NAME" 2>/dev/null || true
    fi

    # Detach policies and delete admin role if it exists
    if [ -n "${ADMIN_ROLE_NAME:-}" ]; then
        echo "Detaching policy from role: $ADMIN_ROLE_NAME"
        aws iam detach-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>/dev/null || true

        for policy in $(aws iam list-role-policies --role-name "$ADMIN_ROLE_NAME"
 --query 'PolicyNames[]' --output text 2>/dev/null || true); do
            aws iam delete-role-policy --role-name "$ADMIN_ROLE_NAME" --policy-
name "$policy" 2>/dev/null || true
        done

        echo "Deleting IAM role: $ADMIN_ROLE_NAME"
        aws iam delete-role --role-name "$ADMIN_ROLE_NAME" 2>/dev/null || true
    fi

    echo "Cleanup completed."
```

```
}

# Trap to ensure cleanup on script exit
trap 'echo "Script interrupted. Running cleanup..."; cleanup_resources' INT TERM
 EXIT

# Generate random identifiers for resources
ADMIN_ROLE_NAME="SecretsManagerAdmin-$(generate_random_id)"
RUNTIME_ROLE_NAME="RoleToRetrieveSecretAtRuntime-$(generate_random_id)"
SECRET_NAME="MyAPIKey-$(generate_random_id)"

echo "Using the following resource names:"
echo "Admin Role: $ADMIN_ROLE_NAME"
echo "Runtime Role: $RUNTIME_ROLE_NAME"
echo "Secret Name: $SECRET_NAME"
echo ""

# Step 1: Create IAM roles
echo "Creating IAM roles..."

# Create assume role policy document
ASSUME_ROLE_POLICY=$(cat <<'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
)

# Create the SecretsManagerAdmin role
echo "Creating admin role: $ADMIN_ROLE_NAME"
ADMIN_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$ADMIN_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$ADMIN_ROLE_OUTPUT" "create-role for admin"
```

```
echo "$ADMIN_ROLE_OUTPUT"
aws iam tag-role --role-name "$ADMIN_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Attach the SecretsManagerReadWrite policy to the admin role
echo "Attaching SecretsManagerReadWrite policy to admin role"
ATTACH_POLICY_OUTPUT=$(aws iam attach-role-policy \
    --role-name "$ADMIN_ROLE_NAME" \
    --policy-arn "arn:aws:iam::aws:policy/SecretsManagerReadWrite" 2>&1)

check_error "$ATTACH_POLICY_OUTPUT" "attach-role-policy for admin"
echo "Policy attached successfully"

# Create the RoleToRetrieveSecretAtRuntime role
echo "Creating runtime role: $RUNTIME_ROLE_NAME"
RUNTIME_ROLE_OUTPUT=$(aws iam create-role \
    --role-name "$RUNTIME_ROLE_NAME" \
    --assume-role-policy-document "$ASSUME_ROLE_POLICY" 2>&1)

check_error "$RUNTIME_ROLE_OUTPUT" "create-role for runtime"
echo "$RUNTIME_ROLE_OUTPUT"
aws iam tag-role --role-name "$RUNTIME_ROLE_NAME" --tags Key=project,Value=doc-
smith Key=tutorial,Value=aws-secrets-manager-gs

# Wait for roles to be fully created
echo "Waiting for IAM roles to be fully created..."
sleep 10

# Step 2: Create a secret in AWS Secrets Manager
echo "Creating secret in AWS Secrets Manager..."

# Generate secure secret value using environment variable or secure method
# WARNING: In production, use secure methods to inject secrets (AWS CodeBuild,
 parameter store, etc.)
if [ -z "${TUTORIAL_SECRET_VALUE:-}" ]; then
    SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_client_id','ClientSecret':__import__('secrets').token_u
else
    SECRET_VALUE="$TUTORIAL_SECRET_VALUE"
fi

CREATE_SECRET_OUTPUT=$(aws secretsmanager create-secret \
    --name "$SECRET_NAME" \
    --description "API key for my application" \
```

```
    --secret-string "$SECRET_VALUE" \
    --add-replica-regions 'Region=us-east-1' \
    --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-secrets-manager-gs
 2>&1)

check_error "$CREATE_SECRET_OUTPUT" "create-secret"
echo "$CREATE_SECRET_OUTPUT"

# Get AWS account ID
echo "Getting AWS account ID..."
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text 2>&1)
check_error "$ACCOUNT_ID" "get-caller-identity"
echo "Account ID: $ACCOUNT_ID"

# Get secret ARN for precise resource policy
echo "Getting secret ARN..."
SECRET_ARN=$(aws secretsmanager describe-secret \
    --secret-id "$SECRET_NAME" \
    --query 'ARN' \
    --output text 2>&1)
check_error "$SECRET_ARN" "describe-secret"

# Add resource policy to the secret with least privilege
echo "Adding resource policy to secret..."
RESOURCE_POLICY=$(cat <<EOF
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRuntimeRoleReadSecret",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::$ACCOUNT_ID:role/$RUNTIME_ROLE_NAME"
            },
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "$SECRET_ARN",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:VersionStage": "AWSCURRENT"
                }
            }
        }
    ]
}
```

```
EOF
)

PUT_POLICY_OUTPUT=$(aws secretsmanager put-resource-policy \
    --secret-id "$SECRET_NAME" \
    --resource-policy "$RESOURCE_POLICY" \
    --block-public-policy 2>&1)

check_error "$PUT_POLICY_OUTPUT" "put-resource-policy"
echo "Resource policy added successfully"

# Enable rotation policy recommendation
echo "Enabling secret metadata tags for rotation tracking..."
aws secretsmanager tag-resource \
    --secret-id "$SECRET_NAME" \
    --tags Key=Purpose,Value=Tutorial Key=AutoRotation,Value=Recommended 2>/dev/
null || true

# Step 3: Demonstrate retrieving the secret
echo "Retrieving the secret value (for demonstration purposes)..."
GET_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$GET_SECRET_OUTPUT" "get-secret-value"
echo "Secret retrieved successfully. Secret metadata:"
echo "$GET_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 4: Update the secret with new values
echo "Updating the secret with new values..."
UPDATE_SECRET_VALUE=$(python3 -c "import json;
 print(json.dumps({'ClientID':'my_new_client_id','ClientSecret':__import__('secrets').tok

UPDATE_SECRET_OUTPUT=$(aws secretsmanager update-secret \
    --secret-id "$SECRET_NAME" \
    --secret-string "$UPDATE_SECRET_VALUE" 2>&1)

check_error "$UPDATE_SECRET_OUTPUT" "update-secret"
echo "Secret updated successfully"

# Step 5: Verify the updated secret
echo "Verifying the updated secret..."
VERIFY_SECRET_OUTPUT=$(aws secretsmanager get-secret-value \
```

```
    --secret-id "$SECRET_NAME" 2>&1)

check_error "$VERIFY_SECRET_OUTPUT" "get-secret-value for verification"
echo "Updated secret retrieved successfully. Secret metadata:"
echo "$VERIFY_SECRET_OUTPUT" | jq '{ARN: .ARN, Name: .Name,
 LastUpdatedDate: .LastUpdatedDate, VersionIdsToStages: .VersionIdsToStages}' 2>/
dev/null || echo "Secret metadata retrieved (jq not available)"

# Step 6: Display rotation recommendations
echo ""
echo "Rotation Configuration Recommendations:"
echo "======================================="
DESCRIBE_OUTPUT=$(aws secretsmanager describe-secret --secret-id "$SECRET_NAME"
 2>&1)
if echo "$DESCRIBE_OUTPUT" | grep -q "RotationRules"; then
    echo "Current rotation configuration:"
    echo "$DESCRIBE_OUTPUT" | jq '.RotationRules' 2>/dev/null || echo "Rotation
 rules available"
else
    echo "No automatic rotation configured. Consider enabling rotation with:"
    echo "aws secretsmanager rotate-secret --secret-id $SECRET_NAME --rotation-
lambda-arn arn:aws:lambda:REGION:ACCOUNT:function:FUNCTION_NAME --rotation-rules
 AutomaticallyAfterDays=30"
fi

echo ""
echo "============================================================"
echo "Tutorial completed successfully!"
echo ""
echo "Summary of what we did:"
echo "1. Created IAM roles for managing and retrieving secrets"
echo "2. Created a secret in AWS Secrets Manager with secure generation"
echo "3. Added a least-privilege resource policy to control access to the secret"
echo "4. Retrieved the secret value (simulating application access)"
echo "5. Updated the secret with cryptographically secure values"
echo "6. Verified the updated secret"
echo ""
echo "Security best practices applied:"
echo "- Used cryptographically secure random ID generation"
echo "- Applied least-privilege resource policies with version stages"
echo "- Tagged resources for rotation tracking"
echo "- Blocked public access to secrets"
echo "- Used ARN-specific permissions instead of wildcards"
echo ""
```

```
echo "Next steps you might want to consider:"
echo "- Enable automatic secret rotation with AWS Lambda"
echo "- Implement secret caching in your application"
echo "- Set up CloudTrail logging for secret access auditing"
echo "- Use AWS CodeGuru Reviewer to find hardcoded secrets in your code"
echo "- For multi-region applications, replicate your secrets across regions"
echo "- Configure VPC endpoints for private access to Secrets Manager"
echo ""

echo "Script completed at $(date)"
exit 0
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateRole
  - CreateSecret
  - DeleteRole
  - DeleteSecret
  - DetachRolePolicy
  - GetCallerIdentity
  - GetSecretValue
  - PutResourcePolicy
  - UpdateSecret

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Executar testes de estresse da CPU em instâncias da máquina virtual usando a injeção de falhas

O código de exemplo abaixo mostra como:

- Criar perfis do IAM.
- Criar um alarme do CloudWatch.
- Criar um modelo de experimento

- Executar o experimento.
- Verificar os resultados.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS FIS CPU Stress Test Tutorial Script
# This script automates the steps in the AWS FIS CPU stress test tutorial

#    approach using epoch time calculations that work across all Linux
 distributions

# Set up logging
LOG_FILE="fis-tutorial-$(date +%Y%m%d-%H%M%S).log"
exec > >(tee -a "$LOG_FILE") 2>&1

echo "Starting AWS FIS CPU Stress Test Tutorial Script"
echo "Logging to $LOG_FILE"
echo "============================================="

# Function to check for errors in command output
check_error() {
    local output=$1
    local cmd=$2

    if echo "$output" | grep -i "error" > /dev/null; then
        # Ignore specific expected errors
        if [[ "$cmd" == *"aws fis get-experiment"* ]] && [[ "$output" ==
 *"ConfigurationFailure"* ]]; then
            echo "Note: Experiment failed due to configuration issue. This is
 expected in some cases."
```

```
            return 0
        fi

        echo "ERROR: Command failed: $cmd"
        echo "Output: $output"
        cleanup_on_error
        exit 1
    fi
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "Error encountered. Cleaning up resources..."

    if [ -n "$EXPERIMENT_ID" ]; then
        echo "Stopping experiment $EXPERIMENT_ID if running..."
        aws fis stop-experiment --id "$EXPERIMENT_ID" 2>/dev/null || true
    fi

    if [ -n "$TEMPLATE_ID" ]; then
        echo "Deleting experiment template $TEMPLATE_ID..."
        aws fis delete-experiment-template --id "$TEMPLATE_ID" || true
    fi

    if [ -n "$INSTANCE_ID" ]; then
        echo "Terminating EC2 instance $INSTANCE_ID..."
        aws ec2 terminate-instances --instance-ids "$INSTANCE_ID" || true
    fi

    if [ -n "$ALARM_NAME" ]; then
        echo "Deleting CloudWatch alarm $ALARM_NAME..."
        aws cloudwatch delete-alarms --alarm-names "$ALARM_NAME" || true
    fi

    if [ -n "$INSTANCE_PROFILE_NAME" ]; then
        echo "Removing role from instance profile..."
        aws iam remove-role-from-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" --role-name "$EC2_ROLE_NAME" || true

        echo "Deleting instance profile..."
        aws iam delete-instance-profile --instance-profile-name
 "$INSTANCE_PROFILE_NAME" || true
    fi
```

```
    if [ -n "$FIS_ROLE_NAME" ]; then
        echo "Deleting FIS role policy..."
        aws iam delete-role-policy --role-name "$FIS_ROLE_NAME" --policy-name
 "$FIS_POLICY_NAME" || true

        echo "Deleting FIS role..."
        aws iam delete-role --role-name "$FIS_ROLE_NAME" || true
    fi

    if [ -n "$EC2_ROLE_NAME" ]; then
        echo "Detaching policy from EC2 role..."
        aws iam detach-role-policy --role-name "$EC2_ROLE_NAME" --policy-arn
 "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" || true

        echo "Deleting EC2 role..."
        aws iam delete-role --role-name "$EC2_ROLE_NAME" || true
    fi

    echo "Cleanup completed."
}

# Generate unique identifiers for resources
TIMESTAMP=$(date +%Y%m%d%H%M%S)
FIS_ROLE_NAME="FISRole-${TIMESTAMP}"
FIS_POLICY_NAME="FISPolicy-${TIMESTAMP}"
EC2_ROLE_NAME="EC2SSMRole-${TIMESTAMP}"
INSTANCE_PROFILE_NAME="EC2SSMProfile-${TIMESTAMP}"
ALARM_NAME="FIS-CPU-Alarm-${TIMESTAMP}"

# Track created resources
CREATED_RESOURCES=()

echo "Step 1: Creating IAM role for AWS FIS"
# Create trust policy file for AWS FIS
cat > fis-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "fis.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
```

```
    }
  ]
}
EOF

# Create IAM role for FIS
echo "Creating IAM role $FIS_ROLE_NAME for AWS FIS..."
FIS_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$FIS_ROLE_NAME" \
  --assume-role-policy-document file://fis-trust-policy.json)
check_error "$FIS_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$FIS_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $FIS_ROLE_NAME")

# Create policy document for SSM actions
cat > fis-ssm-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:SendCommand",
        "ssm:ListCommands",
        "ssm:ListCommandInvocations"
      ],
      "Resource": "*"
    }
  ]
}
EOF

# Attach policy to the role
echo "Attaching policy $FIS_POLICY_NAME to role $FIS_ROLE_NAME..."
FIS_POLICY_OUTPUT=$(aws iam put-role-policy \
  --role-name "$FIS_ROLE_NAME" \
  --policy-name "$FIS_POLICY_NAME" \
  --policy-document file://fis-ssm-policy.json)
check_error "$FIS_POLICY_OUTPUT" "aws iam put-role-policy"
CREATED_RESOURCES+=("IAM Policy: $FIS_POLICY_NAME attached to $FIS_ROLE_NAME")

echo "Step 2: Creating IAM role for EC2 instance with SSM permissions"
# Create trust policy file for EC2
```

```
cat > ec2-trust-policy.json << 'EOF'
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF

# Create IAM role for EC2
echo "Creating IAM role $EC2_ROLE_NAME for EC2 instance..."
EC2_ROLE_OUTPUT=$(aws iam create-role \
  --role-name "$EC2_ROLE_NAME" \
  --assume-role-policy-document file://ec2-trust-policy.json)
check_error "$EC2_ROLE_OUTPUT" "aws iam create-role"
aws iam tag-role --role-name "$EC2_ROLE_NAME" --tags Key=project,Value=doc-smith
 Key=tutorial,Value=aws-fault-injection-service-gs
CREATED_RESOURCES+=("IAM Role: $EC2_ROLE_NAME")

# Attach SSM policy to the EC2 role
echo "Attaching AmazonSSMManagedInstanceCore policy to role $EC2_ROLE_NAME..."
EC2_POLICY_OUTPUT=$(aws iam attach-role-policy \
  --role-name "$EC2_ROLE_NAME" \
  --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore)
check_error "$EC2_POLICY_OUTPUT" "aws iam attach-role-policy"
CREATED_RESOURCES+=("IAM Policy: AmazonSSMManagedInstanceCore attached to
 $EC2_ROLE_NAME")

# Create instance profile
echo "Creating instance profile $INSTANCE_PROFILE_NAME..."
PROFILE_OUTPUT=$(aws iam create-instance-profile \
  --instance-profile-name "$INSTANCE_PROFILE_NAME")
check_error "$PROFILE_OUTPUT" "aws iam create-instance-profile"
CREATED_RESOURCES+=("IAM Instance Profile: $INSTANCE_PROFILE_NAME")

# Add role to instance profile
echo "Adding role $EC2_ROLE_NAME to instance profile $INSTANCE_PROFILE_NAME..."
ADD_ROLE_OUTPUT=$(aws iam add-role-to-instance-profile \
```

```
  --instance-profile-name "$INSTANCE_PROFILE_NAME" \
  --role-name "$EC2_ROLE_NAME")
check_error "$ADD_ROLE_OUTPUT" "aws iam add-role-to-instance-profile"

# Wait for role to propagate
echo "Waiting for IAM role to propagate..."
sleep 10

echo "Step 3: Launching EC2 instance"
# Get the latest Amazon Linux 2 AMI ID
echo "Finding latest Amazon Linux 2 AMI..."
AMI_ID=$(aws ec2 describe-images \
  --owners amazon \
  --filters "Name=name,Values=amzn2-ami-hvm-*-x86_64-gp2"
 "Name=state,Values=available" \
  --query "sort_by(Images, &CreationDate)[-1].ImageId" \
  --output text)
check_error "$AMI_ID" "aws ec2 describe-images"
echo "Using AMI: $AMI_ID"

# Launch EC2 instance
echo "Launching EC2 instance with AMI $AMI_ID..."
INSTANCE_OUTPUT=$(aws ec2 run-instances \
  --image-id "$AMI_ID" \
  --instance-type t2.micro \
  --iam-instance-profile Name="$INSTANCE_PROFILE_NAME" \
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=FIS-Test-
Instance},{Key=project,Value=doc-smith},{Key=tutorial,Value=aws-fault-injection-
service-gs}]')
check_error "$INSTANCE_OUTPUT" "aws ec2 run-instances"

# Get instance ID
INSTANCE_ID=$(echo "$INSTANCE_OUTPUT" | grep -i "InstanceId" | head -1 | awk -
F'"' '{print $4}')
if [ -z "$INSTANCE_ID" ]; then
    echo "Failed to get instance ID"
    cleanup_on_error
    exit 1
fi
echo "Launched instance: $INSTANCE_ID"
CREATED_RESOURCES+=("EC2 Instance: $INSTANCE_ID")

# Enable detailed monitoring
echo "Enabling detailed monitoring for instance $INSTANCE_ID..."
```

```
MONITOR_OUTPUT=$(aws ec2 monitor-instances --instance-ids "$INSTANCE_ID")
check_error "$MONITOR_OUTPUT" "aws ec2 monitor-instances"

# Wait for instance to be running and status checks to pass
echo "Waiting for instance to be ready..."
aws ec2 wait instance-running --instance-ids "$INSTANCE_ID"
aws ec2 wait instance-status-ok --instance-ids "$INSTANCE_ID"
echo "Instance is ready"

echo "Step 4: Creating CloudWatch alarm for CPU utilization"
# Create CloudWatch alarm
echo "Creating CloudWatch alarm $ALARM_NAME..."
ALARM_OUTPUT=$(aws cloudwatch put-metric-alarm \
  --alarm-name "$ALARM_NAME" \
  --alarm-description "Alarm when CPU exceeds 50%" \
  --metric-name CPUUtilization \
  --namespace AWS/EC2 \
  --statistic Maximum \
  --period 60 \
  --threshold 50 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions "Name=InstanceId,Value=$INSTANCE_ID" \
  --evaluation-periods 1)
check_error "$ALARM_OUTPUT" "aws cloudwatch put-metric-alarm"
CREATED_RESOURCES+=("CloudWatch Alarm: $ALARM_NAME")

# Get the alarm ARN
echo "Getting CloudWatch alarm ARN..."
ALARM_ARN_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
check_error "$ALARM_ARN_OUTPUT" "aws cloudwatch describe-alarms"
ALARM_ARN=$(echo "$ALARM_ARN_OUTPUT" | grep -i "AlarmArn" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$ALARM_ARN" ]; then
    echo "Failed to get alarm ARN"
    cleanup_on_error
    exit 1
fi
echo "Alarm ARN: $ALARM_ARN"

# Wait for the alarm to initialize and reach OK state
echo "Waiting for CloudWatch alarm to initialize (60 seconds)..."
sleep 60
```

```
# Check alarm state
echo "Checking alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
  --alarm-names "$ALARM_NAME")
ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 | awk -
F'"' '{print $4}')
echo "Current alarm state: $ALARM_STATE"

# If alarm is not in OK state, wait longer or generate some baseline metrics
if [ "$ALARM_STATE" != "OK" ]; then
    echo "Alarm not in OK state. Waiting for alarm to stabilize (additional 60
 seconds)..."
    sleep 60

    # Check alarm state again
    ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms \
      --alarm-names "$ALARM_NAME")
    ALARM_STATE=$(echo "$ALARM_STATE_OUTPUT" | grep -i "StateValue" | head -1 |
 awk -F'"' '{print $4}')
    echo "Updated alarm state: $ALARM_STATE"

    if [ "$ALARM_STATE" != "OK" ]; then
        echo "Warning: Alarm still not in OK state. Experiment may fail to
 start."
    fi
fi

echo "Step 5: Creating AWS FIS experiment template"
# Get the IAM role ARN
echo "Getting IAM role ARN for $FIS_ROLE_NAME..."
ROLE_ARN_OUTPUT=$(aws iam get-role \
  --role-name "$FIS_ROLE_NAME")
check_error "$ROLE_ARN_OUTPUT" "aws iam get-role"
ROLE_ARN=$(echo "$ROLE_ARN_OUTPUT" | grep -i "Arn" | head -1 | awk -F'"' '{print
 $4}')
if [ -z "$ROLE_ARN" ]; then
    echo "Failed to get role ARN"
    cleanup_on_error
    exit 1
fi
echo "Role ARN: $ROLE_ARN"

# Get account ID and region
ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
```

```
REGION=$(aws configure get region)
if [ -z "$REGION" ]; then
    REGION="us-east-1"  # Default to us-east-1 if region not set
fi
INSTANCE_ARN="arn:aws:ec2:${REGION}:${ACCOUNT_ID}:instance/${INSTANCE_ID}"
echo "Instance ARN: $INSTANCE_ARN"

# Create experiment template - Fixed JSON escaping issue
cat > experiment-template.json << EOF
{
  "description": "Test CPU stress predefined SSM document",
  "targets": {
    "testInstance": {
      "resourceType": "aws:ec2:instance",
      "resourceArns": ["$INSTANCE_ARN"],
      "selectionMode": "ALL"
    }
  },
  "actions": {
    "runCpuStress": {
      "actionId": "aws:ssm:send-command",
      "parameters": {
        "documentArn": "arn:aws:ssm:$REGION::document/AWSFIS-Run-CPU-Stress",
        "documentParameters": "{\"DurationSeconds\":\"120\"}",
        "duration": "PT5M"
      },
      "targets": {
        "Instances": "testInstance"
      }
    }
  },
  "stopConditions": [
    {
      "source": "aws:cloudwatch:alarm",
      "value": "$ALARM_ARN"
    }
  ],
  "roleArn": "$ROLE_ARN",
  "tags": {
    "Name": "FIS-CPU-Stress-Experiment",
    "project": "doc-smith",
    "tutorial": "aws-fault-injection-service-gs"
  }
}
```

```
EOF

# Create experiment template
echo "Creating AWS FIS experiment template..."
TEMPLATE_OUTPUT=$(aws fis create-experiment-template --cli-input-json file://
experiment-template.json)
check_error "$TEMPLATE_OUTPUT" "aws fis create-experiment-template"
TEMPLATE_ID=$(echo "$TEMPLATE_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$TEMPLATE_ID" ]; then
    echo "Failed to get template ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment template created with ID: $TEMPLATE_ID"
CREATED_RESOURCES+=("FIS Experiment Template: $TEMPLATE_ID")

echo "Step 6: Starting the experiment"
# Start the experiment
echo "Starting AWS FIS experiment using template $TEMPLATE_ID..."
EXPERIMENT_OUTPUT=$(aws fis start-experiment \
  --experiment-template-id "$TEMPLATE_ID" \
  --tags '{"Name": "FIS-CPU-Stress-Run"}')
check_error "$EXPERIMENT_OUTPUT" "aws fis start-experiment"
EXPERIMENT_ID=$(echo "$EXPERIMENT_OUTPUT" | grep -i "id" | head -1 | awk -F'"'
 '{print $4}')
if [ -z "$EXPERIMENT_ID" ]; then
    echo "Failed to get experiment ID"
    cleanup_on_error
    exit 1
fi
echo "Experiment started with ID: $EXPERIMENT_ID"
CREATED_RESOURCES+=("FIS Experiment: $EXPERIMENT_ID")

echo "Step 7: Tracking experiment progress"
# Track experiment progress
echo "Tracking experiment progress..."
MAX_CHECKS=30
CHECK_COUNT=0
EXPERIMENT_STATE=""

while [ $CHECK_COUNT -lt $MAX_CHECKS ]; do
    EXPERIMENT_INFO=$(aws fis get-experiment --id "$EXPERIMENT_ID")
    # Don't check for errors here, as we expect some experiments to fail
```

```
    EXPERIMENT_STATE=$(echo "$EXPERIMENT_INFO" | grep -i "status" | head -1 | awk
 -F'"' '{print $4}')
    echo "Experiment state: $EXPERIMENT_STATE"

    if [ "$EXPERIMENT_STATE" == "completed" ] || [ "$EXPERIMENT_STATE" ==
 "stopped" ] || [ "$EXPERIMENT_STATE" == "failed" ]; then
        # Show the reason for the state
        REASON=$(echo "$EXPERIMENT_INFO" | grep -i "reason" | head -1 | awk -F'"'
 '{print $4}')
        if [ -n "$REASON" ]; then
            echo "Reason: $REASON"
        fi
        break
    fi

    echo "Waiting 10 seconds before checking again..."
    sleep 10
    CHECK_COUNT=$((CHECK_COUNT + 1))
done

if [ $CHECK_COUNT -eq $MAX_CHECKS ]; then
    echo "Experiment is taking longer than expected. You can check its status
 later using:"
    echo "aws fis get-experiment --id $EXPERIMENT_ID"
fi

echo "Step 8: Verifying experiment results"
# Check CloudWatch alarm state
echo "Checking CloudWatch alarm state..."
ALARM_STATE_OUTPUT=$(aws cloudwatch describe-alarms --alarm-names "$ALARM_NAME")
check_error "$ALARM_STATE_OUTPUT" "aws cloudwatch describe-alarms"
echo "$ALARM_STATE_OUTPUT"

# Get CPU utilization metrics
echo "Getting CPU utilization metrics..."
END_TIME=$(date -u +"%Y-%m-%dT%H:%M:%SZ")

# FIXED: Cross-platform compatible way to calculate time 10 minutes ago
# This approach uses epoch seconds and basic arithmetic which works on all Linux
 distributions
CURRENT_EPOCH=$(date +%s)
TEN_MINUTES_AGO_EPOCH=$((CURRENT_EPOCH - 600))
```

```
START_TIME=$(date -u -d "@$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ" 2>/dev/
null || date -u -r "$TEN_MINUTES_AGO_EPOCH" +"%Y-%m-%dT%H:%M:%SZ")

# Create metric query file
cat > metric-query.json << EOF
[
  {
    "Id": "cpu",
    "MetricStat": {
      "Metric": {
        "Namespace": "AWS/EC2",
        "MetricName": "CPUUtilization",
        "Dimensions": [
          {
            "Name": "InstanceId",
            "Value": "$INSTANCE_ID"
          }
        ]
      },
      "Period": 60,
      "Stat": "Maximum"
    }
  }
]
EOF

METRICS_OUTPUT=$(aws cloudwatch get-metric-data \
  --start-time "$START_TIME" \
  --end-time "$END_TIME" \
  --metric-data-queries file://metric-query.json)
check_error "$METRICS_OUTPUT" "aws cloudwatch get-metric-data"
echo "CPU Utilization Metrics:"
echo "$METRICS_OUTPUT"

# Display summary of created resources
echo ""
echo "==========================================="
echo "RESOURCES CREATED"
echo "==========================================="
for resource in "${CREATED_RESOURCES[@]}"; do
    echo "- $resource"
done
echo "==========================================="
```

```
# Prompt for cleanup
echo ""
echo "========================================="
echo "CLEANUP CONFIRMATION"
echo "========================================="
echo "Do you want to clean up all created resources? (y/n): "
read -r CLEANUP_CHOICE

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Starting cleanup process..."

    # Stop experiment if still running
    if [ "$EXPERIMENT_STATE" != "completed" ] && [ "$EXPERIMENT_STATE" !=
 "stopped" ] && [ "$EXPERIMENT_STATE" != "failed" ]; then
        echo "Stopping experiment $EXPERIMENT_ID..."
        STOP_OUTPUT=$(aws fis stop-experiment --id "$EXPERIMENT_ID")
        check_error "$STOP_OUTPUT" "aws fis stop-experiment"
        echo "Waiting for experiment to stop..."
        sleep 10
    fi

    # Delete experiment template
    echo "Deleting experiment template $TEMPLATE_ID..."
    DELETE_TEMPLATE_OUTPUT=$(aws fis delete-experiment-template --id
 "$TEMPLATE_ID")
    check_error "$DELETE_TEMPLATE_OUTPUT" "aws fis delete-experiment-template"

    # Delete CloudWatch alarm
    echo "Deleting CloudWatch alarm $ALARM_NAME..."
    DELETE_ALARM_OUTPUT=$(aws cloudwatch delete-alarms --alarm-names
 "$ALARM_NAME")
    check_error "$DELETE_ALARM_OUTPUT" "aws cloudwatch delete-alarms"

    # Terminate EC2 instance
    echo "Terminating EC2 instance $INSTANCE_ID..."
    TERMINATE_OUTPUT=$(aws ec2 terminate-instances --instance-ids "$INSTANCE_ID")
    check_error "$TERMINATE_OUTPUT" "aws ec2 terminate-instances"
    echo "Waiting for instance to terminate..."
    aws ec2 wait instance-terminated --instance-ids "$INSTANCE_ID"

    # Clean up IAM resources
    echo "Removing role from instance profile..."
    REMOVE_ROLE_OUTPUT=$(aws iam remove-role-from-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME" \
```

```
      --role-name "$EC2_ROLE_NAME")
    check_error "$REMOVE_ROLE_OUTPUT" "aws iam remove-role-from-instance-profile"

    echo "Deleting instance profile..."
    DELETE_PROFILE_OUTPUT=$(aws iam delete-instance-profile \
      --instance-profile-name "$INSTANCE_PROFILE_NAME")
    check_error "$DELETE_PROFILE_OUTPUT" "aws iam delete-instance-profile"

    echo "Deleting FIS role policy..."
    DELETE_POLICY_OUTPUT=$(aws iam delete-role-policy \
      --role-name "$FIS_ROLE_NAME" \
      --policy-name "$FIS_POLICY_NAME")
    check_error "$DELETE_POLICY_OUTPUT" "aws iam delete-role-policy"

    echo "Detaching policy from EC2 role..."
    DETACH_POLICY_OUTPUT=$(aws iam detach-role-policy \
      --role-name "$EC2_ROLE_NAME" \
      --policy-arn "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore")
    check_error "$DETACH_POLICY_OUTPUT" "aws iam detach-role-policy"

    echo "Deleting FIS role..."
    DELETE_FIS_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$FIS_ROLE_NAME")
    check_error "$DELETE_FIS_ROLE_OUTPUT" "aws iam delete-role"

    echo "Deleting EC2 role..."
    DELETE_EC2_ROLE_OUTPUT=$(aws iam delete-role \
      --role-name "$EC2_ROLE_NAME")
    check_error "$DELETE_EC2_ROLE_OUTPUT" "aws iam delete-role"

    # Clean up temporary files
    echo "Cleaning up temporary files..."
    rm -f fis-trust-policy.json ec2-trust-policy.json fis-ssm-policy.json
 experiment-template.json metric-query.json

    echo "Cleanup completed successfully."
else
    echo "Cleanup skipped. Resources will remain in your AWS account."
    echo "You can manually clean up the resources listed above."
fi

echo ""
echo "Script execution completed."
echo "Log file: $LOG_FILE"
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AddRoleToInstanceProfile
  - AttachRolePolicy
  - CreateExperimentTemplate
  - CreateInstanceProfile
  - CreateRole
  - DeleteAlarms
  - DeleteExperimentTemplate
  - DeleteInstanceProfile
  - DeleteRole
  - DeleteRolePolicy
  - DescribeAlarms
  - DescribeImages
  - DetachRolePolicy
  - GetCallerIdentity
  - GetExperiment
  - GetMetricData
  - GetRole
  - MonitorInstances
  - PutMetricAlarm
  - PutRolePolicy
  - RemoveRoleFromInstanceProfile
  - RunInstances
  - StartExperiment
  - StopExperiment
  - TerminateInstances
  - Wait

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Como configurar o gerenciamento de sistemas

O código de exemplo abaixo mostra como:

- Criar permissões do IAM para o Systems Manager.
- Criar um perfil do IAM para o Systems Manager
- Configurar o Systems Manager.
- Verificar a configuração.
- Limpar os recursos

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no repositório Sample developer tutorials.

```
#!/bin/bash

# AWS Systems Manager Setup Script
# This script sets up AWS Systems Manager for a single account and region
#
# Version 17 fixes:
# 1. Added cloudformation.amazonaws.com to the IAM role trust policy
# 2. Systems Manager Quick Setup uses CloudFormation for deployments, so the role
 must trust CloudFormation service

# Initialize log file
LOG_FILE="ssm_setup_$(date +%Y%m%d_%H%M%S).log"
UNIQUE_ID=$(openssl rand -hex 4)
echo "Starting AWS Systems Manager setup at $(date)" > "$LOG_FILE"

# Function to log commands and their outputs with immediate terminal display
```

```
log_cmd() {
    echo "$(date): Running command: $1" | tee -a "$LOG_FILE"
    local output
    output=$(eval "$1" 2>&1)
    local status=$?
    echo "$output" | tee -a "$LOG_FILE"
    return $status
}

# Function to check for errors in command output
check_error() {
    local cmd_output="$1"
    local cmd_status="$2"
    local error_msg="$3"

    if [[ $cmd_status -ne 0 || "$cmd_output" =~ [Ee][Rr][Rr][Oo][Rr] ]]; then
        echo "ERROR: $error_msg" | tee -a "$LOG_FILE"
        echo "Command output: $cmd_output" | tee -a "$LOG_FILE"
        cleanup_on_error
        exit 1
    fi
}

# Array to track created resources for cleanup
declare -a CREATED_RESOURCES

# Function to add a resource to the tracking array
track_resource() {
    local resource_type="$1"
    local resource_id="$2"
    CREATED_RESOURCES+=("$resource_type:$resource_id")
    echo "Tracked resource: $resource_type:$resource_id" | tee -a "$LOG_FILE"
}

# Function to clean up resources on error
cleanup_on_error() {
    echo "" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "ERROR OCCURRED - CLEANING UP RESOURCES" | tee -a "$LOG_FILE"
    echo "========================================" | tee -a "$LOG_FILE"
    echo "The following resources were created:" | tee -a "$LOG_FILE"

    # Display resources in reverse order
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
```

```
        echo "${CREATED_RESOURCES[$i]}" | tee -a "$LOG_FILE"
    done

    echo "" | tee -a "$LOG_FILE"
    echo "Attempting to clean up resources..." | tee -a "$LOG_FILE"

    # Clean up resources in reverse order
    cleanup_resources
}

# Function to clean up all created resources
cleanup_resources() {
    # Process resources in reverse order (last created, first deleted)
    for ((i=${#CREATED_RESOURCES[@]}-1; i>=0; i--)); do
        IFS=':' read -r resource_type resource_id <<< "${CREATED_RESOURCES[$i]}"

        echo "Deleting $resource_type: $resource_id" | tee -a "$LOG_FILE"

        case "$resource_type" in
            "IAM_POLICY")
                # Delete the policy (detachment should have been handled when the
 role was deleted)
                log_cmd "aws iam delete-policy --policy-arn $resource_id" || true
                ;;
            "IAM_ROLE")
                # Detach all policies from the role first
                if [[ -n "$POLICY_ARN" ]]; then
                    log_cmd "aws iam detach-role-policy --role-name $resource_id
 --policy-arn $POLICY_ARN" || true
                fi

                # Delete the role
                log_cmd "aws iam delete-role --role-name $resource_id" || true
                ;;
            "SSM_CONFIG_MANAGER")
                log_cmd "aws ssm-quicksetup delete-configuration-manager --
manager-arn $resource_id" || true
                ;;
            *)
                echo "Unknown resource type: $resource_type, cannot delete
 automatically" | tee -a "$LOG_FILE"
                ;;
        esac
    done
```

```
    echo "Cleanup completed" | tee -a "$LOG_FILE"

    # Clean up temporary files
    rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/
null || true
}

# Main script execution
echo "AWS Systems Manager Setup Script"
echo "================================"
echo "This script will set up AWS Systems Manager for a single account and
 region."
echo "It will create IAM policies and roles, then enable Systems Manager
 features."
echo ""

# Get the current AWS region
CURRENT_REGION=$(aws configure get region)
if [[ -z "$CURRENT_REGION" ]]; then
    echo "No AWS region configured. Please specify a region:"
    CURRENT_REGION="${AWS_DEFAULT_REGION:-us-west-2}"
    if [[ -z "$CURRENT_REGION" ]]; then
        echo "ERROR: A region must be specified" | tee -a "$LOG_FILE"
        exit 1
    fi
fi

echo "Using AWS region: $CURRENT_REGION" | tee -a "$LOG_FILE"

# Step 1: Create IAM policy for Systems Manager onboarding
echo "Step 1: Creating IAM policy for Systems Manager onboarding..."

# Create policy document
cat > ssm-onboarding-policy.json << 'EOF'
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "QuickSetupActions",
       "Effect": "Allow",
       "Action": [
         "ssm-quicksetup:*"
       ],
```

```
      "Resource": "*"
    },
    {
      "Sid": "SsmReadOnly",
      "Effect": "Allow",
      "Action": [
        "ssm:DescribeAutomationExecutions",
        "ssm:GetAutomationExecution",
        "ssm:ListAssociations",
        "ssm:DescribeAssociation",
        "ssm:ListDocuments",
        "ssm:ListResourceDataSync",
        "ssm:DescribePatchBaselines",
        "ssm:GetPatchBaseline",
        "ssm:DescribeMaintenanceWindows",
        "ssm:DescribeMaintenanceWindowTasks"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SsmDocument",
      "Effect": "Allow",
      "Action": [
        "ssm:GetDocument",
        "ssm:DescribeDocument"
      ],
      "Resource": [
        "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
        "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
      ]
    },
    {
      "Sid": "SsmEnableExplorer",
      "Effect": "Allow",
      "Action": "ssm:StartAutomationExecution",
      "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
      "Sid": "SsmExplorerRds",
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsSummary",
        "ssm:CreateResourceDataSync",
        "ssm:UpdateResourceDataSync"
```

```
        ],
        "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
      },
      {
        "Sid": "OrgsReadOnly",
        "Effect": "Allow",
        "Action": [
          "organizations:DescribeAccount",
          "organizations:DescribeOrganization",
          "organizations:ListDelegatedAdministrators",
          "organizations:ListRoots",
          "organizations:ListParents",
          "organizations:ListOrganizationalUnitsForParent",
          "organizations:DescribeOrganizationalUnit",
          "organizations:ListAWSServiceAccessForOrganization"
        ],
        "Resource": "*"
      },
      {
        "Sid": "OrgsAdministration",
        "Effect": "Allow",
        "Action": [
          "organizations:EnableAWSServiceAccess",
          "organizations:RegisterDelegatedAdministrator",
          "organizations:DeregisterDelegatedAdministrator"
        ],
        "Resource": "*",
        "Condition": {
          "StringEquals": {
            "organizations:ServicePrincipal": [
              "ssm.amazonaws.com",
              "ssm-quicksetup.amazonaws.com",
              "member.org.stacksets.cloudformation.amazonaws.com",
              "resource-explorer-2.amazonaws.com"
            ]
          }
        }
      },
      {
        "Sid": "CfnReadOnly",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ListStacks",
          "cloudformation:DescribeStacks",
```

```
          "cloudformation:ListStackSets",
          "cloudformation:DescribeOrganizationsAccess"
        ],
        "Resource": "*"
      },
      {
        "Sid": "OrgCfnAccess",
        "Effect": "Allow",
        "Action": [
          "cloudformation:ActivateOrganizationsAccess"
        ],
        "Resource": "*"
      },
      {
        "Sid": "CfnStackActions",
        "Effect": "Allow",
        "Action": [
          "cloudformation:CreateStack",
          "cloudformation:DeleteStack",
          "cloudformation:DescribeStackResources",
          "cloudformation:DescribeStackEvents",
          "cloudformation:GetTemplate",
          "cloudformation:RollbackStack",
          "cloudformation:TagResource",
          "cloudformation:UntagResource",
          "cloudformation:UpdateStack"
        ],
        "Resource": [
          "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
          "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
          "arn:aws:cloudformation:*:*:type/resource/*"
        ]
      },
      {
        "Sid": "CfnStackSetActions",
        "Effect": "Allow",
        "Action": [
          "cloudformation:CreateStackInstances",
          "cloudformation:CreateStackSet",
          "cloudformation:DeleteStackInstances",
          "cloudformation:DeleteStackSet",
          "cloudformation:DescribeStackInstance",
          "cloudformation:DetectStackSetDrift",
          "cloudformation:ListStackInstanceResourceDrifts",
```

```
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStackSet"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*",
        "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
      ]
    },
    {
      "Sid": "ValidationReadonlyActions",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:GetRole"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamRolesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:DeleteRole",
        "iam:GetRole",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
```

```
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "cloudformation.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "IamRolesPoliciesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
```

```
       "Resource": [
         "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
         "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
         "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
         "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
         "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
       ]
     }
   ]
}
EOF

# Create the IAM policy
POLICY_OUTPUT=$(log_cmd "aws iam create-policy --policy-name SSMOnboardingPolicy-
$UNIQUE_ID --policy-document file://ssm-onboarding-policy.json --output json")
POLICY_STATUS=$?
check_error "$POLICY_OUTPUT" $POLICY_STATUS "Failed to create IAM policy"

# Extract the policy ARN
POLICY_ARN=$(echo "$POLICY_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:policy/
SSMOnboardingPolicy-[a-f0-9]*')
if [[ -z "$POLICY_ARN" ]]; then
    echo "ERROR: Failed to extract policy ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created policy
track_resource "IAM_POLICY" "$POLICY_ARN"
aws iam tag-policy --policy-arn "$POLICY_ARN" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created policy: $POLICY_ARN" | tee -a "$LOG_FILE"

# Step 2: Create and configure IAM role for Systems Manager
echo ""
echo "Step 2: Creating IAM role for Systems Manager..."

# Get current user name
```

```
USER_OUTPUT=$(log_cmd "aws sts get-caller-identity --output json")
USER_STATUS=$?
check_error "$USER_OUTPUT" $USER_STATUS "Failed to get caller identity"

# Extract account ID
ACCOUNT_ID=$(echo "$USER_OUTPUT" | grep -o '"Account": "[0-9]*"' | cut -d'"' -f4)
if [[ -z "$ACCOUNT_ID" ]]; then
    echo "ERROR: Failed to extract account ID" | tee -a "$LOG_FILE"
    exit 1
fi

# Generate a unique role name
ROLE_NAME="SSMTutorialRole-$(openssl rand -hex 4)"

# Create trust policy for the role - FIXED: Added cloudformation.amazonaws.com
cat > trust-policy.json << 'EOF'
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com",
                    "ssm-quicksetup.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::ACCOUNT_ID:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF

# Replace ACCOUNT_ID placeholder in trust policy
sed -i "s/ACCOUNT_ID/$ACCOUNT_ID/g" trust-policy.json
```

```
# Create the IAM role
ROLE_OUTPUT=$(log_cmd "aws iam create-role --role-name $ROLE_NAME --assume-role-
policy-document file://trust-policy.json --description 'Role for Systems Manager
 tutorial' --output json")
ROLE_STATUS=$?
check_error "$ROLE_OUTPUT" $ROLE_STATUS "Failed to create IAM role"

# Extract the role ARN
ROLE_ARN=$(echo "$ROLE_OUTPUT" | grep -o 'arn:aws:iam::[0-9]*:role/[^"]*')
if [[ -z "$ROLE_ARN" ]]; then
    echo "ERROR: Failed to extract role ARN" | tee -a "$LOG_FILE"
    cleanup_on_error
    exit 1
fi

# Track the created role
track_resource "IAM_ROLE" "$ROLE_NAME"
aws iam tag-role --role-name "$ROLE_NAME" \
  --tags Key=project,Value=doc-smith Key=tutorial,Value=aws-systems-manager-gs

echo "Created IAM role: $ROLE_NAME" | tee -a "$LOG_FILE"
echo "Role ARN: $ROLE_ARN" | tee -a "$LOG_FILE"

# Set identity variables for cleanup
IDENTITY_TYPE="role"
IDENTITY_NAME="$ROLE_NAME"

# Attach the policy to the role
ATTACH_OUTPUT=$(log_cmd "aws iam attach-role-policy --role-name $ROLE_NAME --
policy-arn $POLICY_ARN")
ATTACH_STATUS=$?
check_error "$ATTACH_OUTPUT" $ATTACH_STATUS "Failed to attach policy to role
 $ROLE_NAME"

echo "Policy attached to role: $ROLE_NAME" | tee -a "$LOG_FILE"

# Step 3: Create Systems Manager configuration using Host Management
echo ""
echo "Step 3: Creating Systems Manager configuration..."

# Generate a random identifier for the configuration name
CONFIG_NAME="SSMSetup-$(openssl rand -hex 4)"

# Create configuration file for Systems Manager setup using Host Management
```

```
# Added both required parameters for single account deployment based on
 CloudFormation documentation
cat > ssm-config.json << EOF
[
  {
    "Type": "AWSQuickSetupType-SSMHostMgmt",
    "LocalDeploymentAdministrationRoleArn": "$ROLE_ARN",
    "LocalDeploymentExecutionRoleName": "$ROLE_NAME",
    "Parameters": {
      "TargetAccounts": "$ACCOUNT_ID",
      "TargetRegions": "$CURRENT_REGION"
    }
  }
]
EOF

echo "Configuration file created:" | tee -a "$LOG_FILE"
cat ssm-config.json | tee -a "$LOG_FILE"

# Create the configuration manager
CONFIG_OUTPUT=$(log_cmd "aws ssm-quicksetup create-configuration-manager --name
 \"$CONFIG_NAME\" --configuration-definitions file://ssm-config.json --region
 $CURRENT_REGION")
CONFIG_STATUS=$?
check_error "$CONFIG_OUTPUT" $CONFIG_STATUS "Failed to create Systems Manager
 configuration"

# Extract the manager ARN
MANAGER_ARN=$(echo "$CONFIG_OUTPUT" | grep -o 'arn:aws:ssm-quicksetup:[^"]*')
if [[ -z "$MANAGER_ARN" ]]; then
    echo "ERROR: Failed to extract manager ARN" | tee -a "$LOG_FILE"
    exit 1
fi

# Track the created configuration manager
track_resource "SSM_CONFIG_MANAGER" "$MANAGER_ARN"

echo "Created Systems Manager configuration: $MANAGER_ARN" | tee -a "$LOG_FILE"

# Step 4: Verify the setup
echo ""
echo "Step 4: Verifying the setup..."

# Wait for the configuration to be fully deployed
```

```
echo "Waiting for the configuration to be deployed (this may take a few
 minutes)..."
sleep 30

# Check the configuration manager status
VERIFY_OUTPUT=$(log_cmd "aws ssm-quicksetup get-configuration-manager --manager-
arn $MANAGER_ARN --region $CURRENT_REGION")
VERIFY_STATUS=$?
check_error "$VERIFY_OUTPUT" $VERIFY_STATUS "Failed to verify configuration
 manager"

echo "Systems Manager setup completed successfully!" | tee -a "$LOG_FILE"

# List the created resources
echo ""
echo "============================================="
echo "CREATED RESOURCES"
echo "============================================="
for resource in "${CREATED_RESOURCES[@]+"${CREATED_RESOURCES[@]}"}"; do
    echo "$resource"
done

# Prompt for cleanup
echo ""
echo "============================================="
echo "CLEANUP CONFIRMATION"
echo "============================================="
echo "Do you want to clean up all created resources? (y/n): "
CLEANUP_CHOICE="y"

if [[ "$CLEANUP_CHOICE" =~ ^[Yy]$ ]]; then
    echo "Cleaning up resources..." | tee -a "$LOG_FILE"
    cleanup_resources
    echo "Cleanup completed." | tee -a "$LOG_FILE"
else
    echo "Resources will not be cleaned up. You can manually clean them up
 later." | tee -a "$LOG_FILE"
fi

echo ""
echo "Script execution completed. See $LOG_FILE for details."

# Clean up temporary files
```

```
rm -f ssm-onboarding-policy.json trust-policy.json ssm-config.json 2>/dev/null ||
 true
```

- Consulte detalhes da API nos tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateConfigurationManager
  - CreatePolicy
  - CreateRole
  - DeleteConfigurationManager
  - DeletePolicy
  - DeleteRole
  - DetachRolePolicy
  - GetCallerIdentity
  - GetConfigurationManager

Para ver uma lista completa dos guias de desenvolvedor e exemplos de código do SDK da AWS, consulte Using this service with an AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Segurança no IAM e no AWS STS

A segurança na nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você se beneficiará de data centers e arquiteturas de rede criados para atender aos requisitos das empresas com as maiores exigências de segurança.

A segurança é uma responsabilidade compartilhada entre a AWS e você. O modelo de responsabilidade compartilhada descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem: a AWS é responsável pela proteção da infraestrutura que executa serviços AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. Auditores terceirizados testam e verificam regularmente a eficácia da nossa segurança como parte dos Programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao AWS Identity and Access Management (IAM), consulte Produtos da AWS no escopo por programa de conformidade.
- Segurança na nuvem: sua responsabilidade é determinada pelo serviço da AWS que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da sua empresa e as leis e regulamentos aplicáveis.

Esta documentação ajuda a entender como aplicar o modelo de responsabilidade compartilhada ao usar o AWS Identity and Access Management (IAM) e o AWS Security Token Service (AWS STS). Os tópicos a seguir mostram como configurar o IAM e o AWS STS para atender aos seus objetivos de segurança e de conformidade. Saiba também como usar outros produtos da AWS que ajudam a monitorar e proteger os recursos do IAM.

Conteúdo

- AWSCredenciais de segurança de
- AWSDiretrizes de auditoria de segurança da
- Proteção de dados no AWS Identity and Access Management
- Registrar em log e monitorar no AWS Identity and Access Management
- Validação de conformidade do AWS Identity and Access Management
- Resiliência no AWS Identity and Access Management
- Segurança da infraestrutura no AWS Identity and Access Management
- Análise de vulnerabilidade e configuração no AWS Identity and Access Management

- Políticas da AWS gerenciadas para AWS Identity and Access Management, IAM Access Analyzer e gerenciador de acesso à conta
- Recursos de segurança fora do IAM

AWSCredenciais de segurança de

Ao interagir com a AWS, você especifica as credenciais de segurança da AWS para verificar quem você é e se tem permissão para acessar os recursos que está solicitando. A AWS usa as credenciais de segurança para autenticar e autorizar suas solicitações.

Por exemplo, se você quiser baixar um arquivo protegido de um bucket do Amazon Simple Storage Service (Amazon S3), suas credenciais devem permitir esse acesso. Se suas credenciais não mostram que você tem autorização para baixar o arquivo, a AWS nega sua solicitação. Porém, suas credenciais de segurança da AWS não são obrigatórias para baixar um arquivo em um bucket do Amazon S3 que seja compartilhado publicamente.

Existem diferentes tipos de usuários da AWS, cada um com suas próprias credenciais de segurança:

- Proprietário da conta (usuário-raiz): o usuário que criou a Conta da AWS e tem acesso total.
- Usuários do Centro de Identidade do AWS IAM: usuários gerenciados no Centro de Identidade do AWS IAM.
- Entidades principais federadas: usuários de provedores de identidades externos aos quais é concedido acesso temporário à AWS por meio de federação. Para obter mais informações sobre identidades federadas, consulte Federação e provedores de identidade na AWS.
- Usuários do IAM: usuários individuais criados dentro do serviço AWS Identity and Access Management (IAM).

Os usuários têm credenciais de segurança temporárias ou de longo prazo. O usuário-raiz, o usuário do IAM e as chaves de acesso têm credenciais de segurança de longo prazo que não expiram. Para proteger as credenciais de longo prazo, tenha processos em vigor para gerenciar chaves de acesso, alterar senhas e habilitar a MFA.

Para simplificar o gerenciamento das credenciais de usuário-raiz em todas as contas-membro no AWS Organizations, é possível proteger centralmente as credenciais de usuário-raiz das suas Contas da AWS gerenciadas usando o AWS Organizations. Gerencie centralmente o acesso raiz para contas-membro permite que você remova e evite centralmente a recuperação de credenciais de usuário-raiz em longo prazo, evitando o acesso raiz não intencional em grande escala.

Perfis do IAM, usuários no Centro de Identidade do AWS IAM e entidades principais de usuários federados do AWS STS têm credenciais de segurança temporárias. As credenciais de segurança temporárias expiram após um período definido ou quando o usuário encerra a sessão. As credenciais temporárias funcionam quase de forma idêntica às credenciais de longo prazo, com as seguintes diferenças:

- As credenciais de segurança temporárias são de curto prazo, como o nome indica. Elas podem ser configuradas para durar de alguns minutos a várias horas. Depois que as credenciais expiram, a AWS não as reconhece mais ou permite qualquer tipo de acesso de solicitações de API feitas com elas.
- As credenciais de segurança temporárias não são armazenadas com o usuário, mas são geradas dinamicamente e fornecidas ao usuário quando solicitadas. Quando (ou até mesmo antes) as credenciais de segurança temporárias expiram, o usuário pode solicitar novas credenciais, desde que o usuário solicitante ainda tenha permissões para fazê-lo.

Como resultado, as credenciais temporárias apresentam as seguintes vantagens em relação às credenciais de longo prazo:

- Você não tem que distribuir ou incorporar credenciais de segurança da AWS de longo prazo com um aplicativo.
- É possível fornecer acesso aos seus recursos da AWS para os usuários sem a necessidade de definir uma identidade da AWS para eles. As credenciais temporárias são a base para funções e federação de identidades.
- As credenciais de segurança temporárias têm vida limitada. Portanto, não é necessário atualizá-las ou explicitamente revogá-las quando elas não forem mais necessárias. Quando as credenciais de segurança temporárias expiram, elas não podem ser reutilizadas. É possível especificar por quanto tempo as credenciais são válidas, até um limite máximo.

Considerações sobre segurança

Recomendamos considerar as seguintes informações ao determinar as provisões de segurança para sua Conta da AWS:

- Quando você cria uma Conta da AWS, nós criamos a conta de usuário-raiz. As credenciais do usuário-raiz (proprietário da conta) permitem acesso total a todos os recursos da conta. A primeira

tarefa que você executa com o usuário-raiz é conceder a outro usuário permissões administrativas para sua Conta da AWS para minimizar o uso do usuário-raiz.

- A autenticação multifator (MFA) fornece um nível adicional de segurança para usuários que podem acessar sua Conta da AWS. Para reforçar a segurança, recomendamos que você exija a MFA nas credenciais de Usuário raiz da conta da AWS e em todos os usuários do IAM. Para obter mais informações, consulte AWSCódigo da autenticação multifator no IAM da.
- A AWS exige diferentes tipos de credenciais de segurança, dependendo de como você acessa a AWS e do tipo de usuário da AWS que você é. Por exemplo, você usa credenciais de login para o Console de gerenciamento da AWS e usa chaves de acesso para fazer chamadas programáticas para a AWS. Para obter ajuda para determinar o tipo de usuário e a página de login, consulte What is AWS Sign-In no Guia do usuário do Início de Sessão da AWS.
- Não é possível usar as políticas do IAM para negar explicitamente ao usuário-raiz o acesso aos recursos. Só é possível usar uma política de controle de serviços (SCP) do AWS Organizations para limitar as permissões do usuário-raiz.
- Caso esqueça ou perca sua senha de usuário-raiz, você deverá ter acesso ao endereço de e-mail associado à conta para redefini-la.
- Caso perca suas chaves de acesso de usuário-raiz, você deverá conseguir fazer login na conta como usuário-raiz para criar novas.
- Não use o usuário-raiz para tarefas cotidianas. Use para executar as tarefas que somente o usuário-raiz pode executar. Para obter a lista completa das tarefas que exigem fazer login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz.
- As credenciais de segurança são específicas para cada conta. Se tiver acesso a várias Contas da AWS, você terá credenciais separadas para cada conta.
- As políticas determinam quais ações um usuário, um perfil ou o membro de um grupo de usuários pode executar, em quais recursos da AWS e em quais condições. Ao usar políticas, você pode controlar com segurança o acesso a Serviços da AWS e recursos em sua Conta da AWS. Se precisar modificar ou revogar as permissões em resposta a um evento de segurança, exclua ou modifique as políticas em vez de fazer alterações diretamente na identidade.
- Salve as credenciais de login de seu usuário do IAM de acesso de emergência e todas as chaves de acesso que você criou para acesso programático em um local seguro. Caso perca suas chaves de acesso, você deverá fazer login na conta para criar novas.
- É altamente recomendável usar credenciais temporárias fornecidas por perfis do IAM e entidades principais federadas em vez das credenciais de longo prazo fornecidas por usuários do IAM e chaves de acesso.

Acesso programático com credenciais de segurança da AWS

Recomendamos usar chaves de acesso de curto prazo quando possível para fazer chamadas programáticas para a AWS ou usar a AWS Command Line Interface ou o Ferramentas da AWS para PowerShell. No entanto, você também pode usar chaves de acesso da AWS de longo prazo para essas finalidades.

Ao criar uma chave de acesso de longo prazo, você cria o ID de chave de acesso (por exemplo, `AKIAIOSFODNN7EXAMPLE`) e a chave de acesso secreta (por exemplo, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) como um conjunto. A chave de acesso secreta só está disponível para baixar no momento em que é criada. Se não fizer download da chave de acesso secreta ou perdê-la, você deverá criar uma nova.

Em muitos casos, você não precisa de chaves de acesso de longo prazo que nunca expiram (como é necessário ao criar chaves de acesso para um usuário do IAM). Em vez disso, você pode criar perfis do IAM e gerar credenciais de segurança temporárias. As credenciais de segurança temporárias incluem um ID da chave de acesso e uma chave de acesso secreta, mas também incluem um token de segurança que indica quando as credenciais expiram. Depois que expiram, não são mais válidas. Para obter mais informações, consulte . Alternativas para chaves de acesso de longo prazo

Os IDs da chave de acesso que começam com `AKIA` são chaves de acesso de longo prazo para um usuário do IAM ou um usuário raiz da Conta da AWS. Os IDs de chave de acesso que começam com `ASIA` são chaves de acesso temporárias que você cria usando operações do AWS STS.

Os usuários precisam de acesso programático se quiserem interagir com a AWS de fora do Console de gerenciamento da AWS. A forma de conceder acesso programático depende do tipo de usuário que está acessando a AWS.

Para conceder acesso programático aos usuários, selecione uma das seguintes opções:

Qual usuário precisa de acesso programático?	Para	Por
IAM	(Recomendado) Use credenciais do console como credenciais temporárias para assinar solicitações programáticas para a AWS CLI, os AWS SDKs ou as APIs da AWS.	Siga as instruções da interface que deseja utilizar. • Para a AWS CLI, consulte Login para desenvolvimento local da AWS no Guia do

Qual usuário precisa de acesso programático?	Para	Por
		usuário do AWS Command Line Interface. • Para AWS SDKs, consulte Login para desenvolvimento local da AWS no Guia de referência de AWS SDKs e ferramentas.
Identidade da força de trabalho (Usuários gerenciados no Centro de Identidade do IAM)	Use credenciais temporárias para assinar solicitações programáticas para a AWS CLI, os SDKs da AWS ou as APIs da AWS.	Siga as instruções da interface que deseja utilizar. • Para a AWS CLI, consulte Configuração da AWS CLI para usar o Centro de Identidade do AWS IAM no Guia do usuário da AWS Command Line Interface. • Para os SDKs da AWS, ferramentas e APIs da AWS, consulte Autenticação do Centro de Identidade do IAM no Guia de referência de ferramentas e SDKs da AWS.
IAM	Use credenciais temporárias para assinar solicitações programáticas para a AWS CLI, os SDKs da AWS ou as APIs da AWS.	Siga as instruções em Usar credenciais temporárias com recursos da AWS no Guia do usuário do IAM.

Qual usuário precisa de acesso programático?	Para	Por
IAM	(Não recomendado) Use credenciais de longo prazo para assinar solicitaç ões programáticas para a AWS CLI, os SDKs da AWS ou as APIs da AWS.	Siga as instruções da interface que deseja utilizar. • Para a AWS CLI, consulte Autenticação usando as credenciais de usuário do IAM no Guia do usuário da AWS Command Line Interface. • Para as ferramentas e SDKs da AWS, consulte Autenticação usando as credenciais de longo prazo no Guia de referência de ferramentas e SDKs da AWS. • Para as APIs da AWS, consulte Gerenciamento de chaves de acesso de usuários do IAM no Guia do usuário do IAM.

Alternativas para chaves de acesso de longo prazo

Para muitos casos de uso comuns, há alternativas às chaves de acesso de longo prazo. Para melhorar a segurança de sua conta, considere as instruções a seguir.

- Não incorpore chaves de acesso de longo prazo e chaves de acesso secretas ao código de sua aplicação ou a um repositório de código: em vez disso, use o AWS Secrets Manager ou outra solução de gerenciamento de segredos para não precisar codificar chaves em texto não criptografado. A aplicação ou o cliente pode então recuperar segredos quando necessário. Para obter mais informações, consulte O que é o AWS Secrets Manager? no Guia do usuário do AWS Secrets Manager.

- Use perfis do IAM para gerar credenciais de segurança temporárias sempre que possível: use sempre mecanismos para emitir credenciais de segurança temporárias quando possível, em vez de chaves de acesso de longo prazo. As credenciais de segurança temporárias são mais seguras porque não são armazenadas com o usuário, mas são geradas dinamicamente e fornecidas ao usuário quando solicitadas. Como as credenciais de segurança temporárias têm uma vida útil limitada, não é necessário gerenciá-las ou atualizá-las. Os mecanismos que fornecem chaves de acesso temporárias incluem perfis do IAM ou a autenticação de um usuário do Centro de Identidade do IAM. Para máquinas que funcionam fora da AWS, você pode usar o AWS Identity and Access Management Roles Anywhere.
- Use alternativas às chaves de acesso de longo prazo para a AWS Command Line Interface (AWS CLI) ou **`aws-shell`**: as alternativas incluem as opções a seguir.
  - Login para desenvolvimento local da AWS usando credenciais do console. Você pode usar a versão 2 da AWS CLI e o comando `aws login` para gerar credenciais de curto prazo para executar comandos da AWS CLI usando as credenciais do console. Para saber mais, consulte Login para desenvolvimento local da AWS no Guia do usuário da AWS Command Line Interface.
  - O AWS CloudShell é um shell pré-autenticado que você pode iniciar diretamente do Console de gerenciamento da AWS. Você pode executar comandos da AWS CLI em Serviços da AWS usando o shell de sua preferência (Bash, Powershell ou Z shell). Ao fazer isso, você não precisa baixar nem instalar ferramentas de linha de comando. Para obter mais informações, consulte O que é o AWS CloudShell? no Guia do usuário do AWS CloudShell.
- Não crie chaves de acesso de longo prazo para usuários humanos que precisam acessar aplicações ou Serviços da AWS: o Centro de Identidade do IAM pode gerar credenciais de acesso temporárias para que seus usuários de IdP externo acessem os Serviços da AWS. Isso elimina a necessidade de criar e gerenciar credenciais de longo prazo no IAM. No Centro de Identidade do IAM, crie um conjunto de permissões do Centro de Identidade IAM que conceda acesso aos usuários de IdP externo. Em seguida, atribua um grupo do Centro de Identidade IAM ao conjunto de permissões nas selecionadas Contas da AWS. Para obter mais informações, consulte O que é o Centro de Identidade do AWS IAM, Conectar-se a um provedor de identidades externo e Conjuntos de permissões no Guia do usuário do Centro de Identidade do AWS IAM.
- Não armazene chaves de acesso de longo prazo em um serviço de computação da AWS: em vez disso, atribua um perfil do IAM aos recursos de computação. Isso automaticamente fornece credenciais temporárias para conceder acesso. Por exemplo, ao criar um perfil de instância que esteja anexado a uma instância do Amazon EC2, você pode atribuir um perfil da AWS à instância e disponibilizá-la para todas as suas aplicações. Um perfil de instância contém o perfil e permite que programas que estejam em execução na instância do Amazon EC2 obtenham credenciais

temporárias. Para saber mais, consulte Uso de um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2.

AWSDiretrizes de auditoria de segurança da

Audite sua configuração de segurança periodicamente para garantir que ela atenda às suas necessidades de negócios atuais. Uma auditoria oferece a você a oportunidade de remover usuários, perfis, grupos e políticas do IAM desnecessários e de garantir que seus usuários e o software não tenham permissões excessivas.

Veja a seguir diretrizes para analisar e monitorar sistematicamente seus recursos da AWS quanto às melhores práticas de segurança.

> **Tip**
>
> É possível monitorar seu uso do IAM em relação às práticas recomendadas de segurança com o AWS Security Hub CSPM. O CSPM do Security Hub usa controles de segurança para avaliar configurações de recursos e padrões de segurança que ajudam você a atender a vários frameworks de conformidade. Para obter mais informações sobre como usar o Security Hub CSPM para avaliar os recursos do IAM, consulte Controles do AWS Identity and Access Management no Guia do usuário do AWS Security Hub CSPM.

Conteúdo

- Quando realizar uma auditoria de segurança
- Diretrizes para a auditoria
- Analisar as credenciais da sua conta da AWS
- Revisar seus usuários do IAM
- Revisar seus grupos do IAM
- Revisar seus perfis do IAM
- Revisar seus provedores do IAM para SAML e OpenID Connect (OIDC)
- Analisar os aplicativos móveis
- Dicas para revisar políticas do IAM

Quando realizar uma auditoria de segurança

Audite sua configuração de segurança nas seguintes situações:

- Periodicamente. Como melhor prática de segurança, execute as etapas descritas neste documento em intervalos regulares.
- Se houver alterações na sua organização, como demissões.
- Se você parou de usar um ou mais serviços da AWS individuais para verificar se removeu permissões das quais os usuários da sua conta não precisam mais.
- Se você tiver adicionado ou removido software de suas contas, como aplicações em instâncias do Amazon EC2, pilhas do OpsWorks, modelos do CloudFormation etc.
- Se você suspeitar de que uma pessoa não autorizada possa ter acesso à sua conta.

Diretrizes para a auditoria

À medida que você examina a configuração de segurança da sua conta, siga estas diretrizes:

- Seja minucioso. Examine todos os aspectos da configuração de segurança, incluindo aqueles raramente usados.
- Não suponha nada. Se você não estiver familiarizado com alguns aspectos da sua configuração de segurança (por exemplo, o raciocínio referente a uma política específica ou a existência de uma função), investigue a necessidade da empresa até entender o risco potencial.
- Simplifique as coisas. Para facilitar a auditoria (e o gerenciamento), use grupos do IAM, perfis do IAM, esquemas de nomenclatura consistentes e políticas claras.

Analisar as credenciais da sua conta da AWS

Ao auditar suas credenciais da conta da AWS, execute estas etapas:

1. Caso tenha chaves de acesso do seu usuário raiz que não estão sendo usadas, você poderá removê-las. É altamente recomendável não usar chaves de acesso raiz para o trabalho cotidiano com a AWS. Em vez disso, utilize usuários com credenciais temporárias, como usuários no Centro de Identidade do AWS IAM.
2. Caso precise de chaves de acesso para sua conta, certifique-se de atualizá-las quando necessário.

Revisar seus usuários do IAM

Ao auditar seus usuários do IAM, execute estas etapas:

1. Liste os usuários e exclua os usuários desnecessários.
2. Remova usuários de grupos aos quais eles não precisam de acesso.
3. Analise as políticas associadas aos grupos nos quais o usuário está. Consulte Dicas para revisar políticas do IAM.
4. Exclua credenciais de segurança que o usuário não precisa ou que podem ter sido expostas. Por exemplo, um usuário do IAM usado para uma aplicação não precisa de senha (que é necessária apenas para fazer login em sites da AWS). Da mesma forma, se um usuário não utiliza chaves de acesso, não há motivo para que ele tenha uma. Para obter mais informações, consulte Gerenciamento de senhas de usuários do IAM e Gerenciamento de chaves de acesso de usuários do IAM.

   Você pode gerar e baixar um relatório de credenciais que lista todos os usuários do IAM em sua conta e o status de diversas credenciais deles, incluindo senhas, chaves de acesso e dispositivos com MFA. Em caso de senhas e chaves de acesso, o relatório de credenciais mostra a data e a hora em que a senha ou chave de acesso foi usada pela última vez. Considere remover da sua conta credenciais que não foram usadas recentemente. (Não remova o usuário de acesso de emergência.) Para obter mais informações, consulte Obter relatórios de credenciais da sua conta da AWS.
5. Atualize senhas e chaves de acesso quando necessário para casos de uso que exijam credenciais de longo prazo. Para obter mais informações, consulte Gerenciamento de senhas de usuários do IAM e Gerenciamento de chaves de acesso de usuários do IAM.
6. Como prática recomendada, exija que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias. Se possível, faça a transição de usuários do IAM para entidades principais federadas, por exemplo, usuários no Centro de Identidade do IAM. Mantenha o número mínimo de usuários do IAM necessário para suas aplicações.

Revisar seus grupos do IAM

Ao auditar seus grupos do IAM, execute estas etapas:

1. Liste seus grupos e exclua os grupos que não está usando.

2. Analise os usuários em cada grupo e remova os usuários que não fizerem parte deles.
3. Analise as políticas associadas ao grupo. Consulte Dicas para revisar políticas do IAM.

Revisar seus perfis do IAM

Ao auditar seus perfis do IAM, execute estas etapas:

1. Liste seus perfis e exclua os perfis que não está usando.
2. Analise a política de confiança da função. Certifique-se de que você sabe quem é o “principal” e que você entende por que essa conta ou usuário precisa ser capaz de assumir a função.
3. Analise a política de acesso da função para garantir que ela concede permissões adequadas para quem assumir a função – consulte Dicas para revisar políticas do IAM.

Revisar seus provedores do IAM para SAML e OpenID Connect (OIDC)

Se você tiver criado uma entidade do IAM para estabelecer confiança com um provedor de identidade (IdP) SAML ou OIDC, faça o seguinte:

1. Exclua provedores não utilizados.
2. Faça o download e analise os documentos de metadados da AWS de cada IdP SAML e certifique-se de que os documentos espelhem suas necessidades de negócios atuais.
3. Obtenha os documentos de metadados mais recentes dos IdPs SAML e atualize o provedor no IAM.

Analisar os aplicativos móveis

Se você tiver criado um aplicativo para dispositivos móveis que faça solicitações para a AWS, execute estas etapas:

1. Certifique-se de que a aplicação para dispositivos móveis não contenham chaves de acesso incorporadas, mesmo se elas estiverem em armazenamento criptografado.
2. Obtenha credenciais temporárias para o aplicativo usando APIs desenvolvidas para essa finalidade.

> **Note**
>
> Recomendamos usar o Amazon Cognito para gerenciar a identidade de usuários em sua aplicação. Esse serviço permite autenticar os usuários usando login com o Amazon, Facebook, Google ou qualquer provedor de identidade compatível com o OpenID Connect (OIDC). Para obter mais informações, consulte Grupos de identidade do Amazon Cognito no Guia do desenvolvedor do Amazon Cognito.

Dicas para revisar políticas do IAM

As políticas são poderosas e sutis. Por isso, é importante analisar e compreender as permissões concedidas por elas. Ao analisar políticas, use as diretrizes a seguir:

- Anexe políticas a grupos ou perfis, e não a usuários individuais. Se um usuário específico tiver uma política, certifique-se de que você entende por que esse usuário precisa da política.
- Certifique-se de que os usuários, grupos e funções do IAM tenham somente as permissões necessárias e nenhuma permissão adicional.
- Use o simulador de políticas do IAM para testar políticas anexadas a usuários ou grupos.
- Lembre-se de que as permissões de um usuário são o resultado de todas as políticas aplicáveis: políticas baseadas em identidade (para usuários, grupos ou perfis) e políticas baseadas em recursos (em recursos como buckets do Amazon S3, filas do Amazon SQS, tópicos do Amazon SNS e chaves do AWS KMS). É importante examinar todas as políticas que se aplicam a um usuário e compreender o conjunto completo de permissões concedidas a um usuário específico.
- Lembre-se de que, ao permitir que um usuário crie um usuário, grupo, perfil ou política do IAM e associe uma política à entidade principal, estão sendo concedidas efetivamente a esse usuário todas as permissões para todos os recursos em sua conta. Os usuários que têm permissão para criar políticas e associá-las a um usuário, grupo ou perfil podem conceder quaisquer permissões por eles mesmos. Em geral, não conceda permissões do IAM com acesso total aos recursos de sua conta a usuários ou perfis em quem não confia. Ao realizar sua auditoria de segurança, verifique se as seguintes permissões do IAM foram concedidas a identidades confiáveis:
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`

  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Certifique-se de que as políticas não concedem permissões para serviços que você não utiliza. Por exemplo, se você usar políticas gerenciadas da AWS, certifique-se de que as políticas gerenciadas da AWS que estão sendo usadas em sua conta são para serviços que você usa realmente. Para saber quais políticas gerenciadas da AWS são usadas em sua conta, use o comando de API `GetAccountAuthorizationDetails` do IAM (comando da AWS CLI: `aws iam get-account-authorization-details`).
- Se a política conceder permissão a um usuário para iniciar uma instância do Amazon EC2, ela também poderá permitir a ação `iam:PassRole`, mas, nesse caso, deverá listar explicitamente os perfis que o usuário poderá passar para a instância do Amazon EC2.
- Examine atentamente todos os valores do elemento `Action` ou `Resource` que inclui `*`. Quando possível, conceda acesso `Allow` às ações e recursos individuais de que os usuários precisam. No entanto, as razões pelas quais pode ser adequado usar `*` em uma política são:
  - A política foi desenvolvida para conceder permissões de nível administrativo.
  - O caractere curinga é usado para um conjunto de ações semelhantes (por exemplo, `Describe*`) como uma facilidade e você está familiarizado com a lista completa de ações que são referenciadas dessa forma.
  - O caractere curinga é usado para indicar uma classe de recursos ou um caminho de recursos (por exemplo, `arn:aws:iam::account-id:users/division_abc/*`) e você se sente à vontade para conceder acesso a todos os recursos na classe ou caminho.
  - Uma ação de serviço não oferece suporte a permissões em nível de recursos, e a única opção para um recurso é `*`.
- Examine os nomes de políticas para garantir que eles espelham a função da política. Por exemplo, embora uma política possa ter um nome que inclui "somente leitura", a política pode, na verdade, conceder permissões de escrita ou de alteração.

Para obter mais informações sobre o planejamento de sua auditoria de segurança, consulte Práticas recomendadas de segurança, identidade e conformidade na Central de arquitetura da AWS.

Proteção de dados no AWS Identity and Access Management

O AWS modelo de responsabilidade compartilhada se aplica à proteção de dados no AWS Identity and Access Management. Conforme descrito nesse modelo, a AWS é responsável por proteger a infraestrutura global que executa toda a Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para ter mais informações sobre privacidade de dados, consulte as Perguntas frequentes sobre privacidade de dados. Para obter informações sobre proteção de dados na Europa, consulte o Centro de Regulamento Geral sobre a Proteção de Dados (RGPD).

Para fins de proteção de dados, recomendamos que você proteja as credenciais da Conta da AWS e configure as contas de usuário individuais com Centro de Identidade do AWS IAM ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos da AWS. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure os logs de API e atividade do usuário com AWS CloudTrail. Para saber mais sobre como usar as trilhas do CloudTrail para capturar atividades da AWS, consulte Working with CloudTrail trails no Guia do usuário do AWS CloudTrail.
- Use as soluções de criptografia AWS, juntamente com todos os controles de segurança padrão em Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sensíveis armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar a AWS por meio de uma interface de linha de comandos ou de uma API, use um endpoint do FIPS. Para saber mais sobre os endpoints FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-3.

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sensíveis, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui trabalhar com o IAM ou outros Serviços da AWS usando o console, a API, a AWS CLI ou os SDKs da AWS. Quaisquer dados inseridos em tags ou em campos de texto de formato livre

usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

Criptografia de dados no IAM e no AWS STS

A criptografia de dados geralmente se encaixa em duas categorias: criptografia em repouso e criptografia em trânsito.

Criptografia em repouso

Os dados coletados e armazenados pelo IAM são criptografados em repouso.

- IAM: os dados coletados e armazenados no IAM incluem endereços IP, metadados da conta do cliente e dados de identificação do cliente que incluem senhas. Os metadados da conta do cliente e os dados de identificação do cliente são criptografados em repouso usando o AES 256 e o SHA 256 com hash.
- AWS STS – AWS STS o não coleta conteúdo do cliente, exceto para logs de serviço que registram solicitações bem-sucedidas, com erro e com falha para o serviço.

Criptografia em trânsito

Os dados de identificação do cliente, incluindo senhas, são criptografados em trânsito usando o TLS 1.2 e 1.3. Todos os endpoints do AWS STS oferecem suporte a HTTPS para criptografar dados em trânsito. Para obter uma lista de endpoints do AWS STS, consulte AWS STSRegiões e endpoints do.

Gerenciamento de chaves no IAM e no AWS STS

Não é possível gerenciar chaves de criptografia usando o IAM ou o AWS STS. Para obter mais informações sobre chaves de criptografia, consulte O que é o AWS KMS? no Guia do desenvolvedor do AWS Key Management Service

Privacidade do tráfego entre redes no IAM e no AWS STS

As solicitações ao IAM devem ser feitas usando o protocolo Transport Layer Security (TLS). É possível proteger conexões com o serviço do AWS STS usando VPC endpoints. Para saber mais, consulte Endpoints da VPC de interface.

Registrar em log e monitorar no AWS Identity and Access Management

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e performance do AWS Identity and Access Management (IAM), do AWS Security Token Service (AWS STS) e de suas outras soluções da AWS. A AWS fornece várias ferramentas para monitorar seus recursos da AWS e responder a potenciais incidentes:

- O AWS CloudTrail captura todas as chamadas de API para o IAM e o AWS STS como eventos, incluindo chamadas do console e chamadas de API. Para saber mais sobre como usar o CloudTrail com o IAM e o AWS STS, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail. Para obter mais informações sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.
- O AWS Identity and Access Management and Access Analyzer ajuda você a identificar os recursos em sua organização e suas contas, como buckets do Amazon S3 ou perfis do IAM, que são compartilhados com uma entidade externa. Isso ajuda a identificar o acesso não intencional aos seus recursos e dados, o que é um risco de segurança. Para saber mais, consulte O que é o IAM Access Analyzer?
- O Amazon CloudWatch monitora os recursos da AWS e as aplicações que você executa na AWS em tempo real. Você pode coletar e rastrear métricas, criar painéis personalizados e definir alarmes que o notificam ou que realizam ações quando uma métrica especificada atinge um limite definido. Por exemplo, você pode fazer o CloudWatch acompanhar o uso da CPU ou outras métricas das instâncias do Amazon EC2 e iniciar automaticamente novas instâncias quando necessário. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch.
- O Amazon CloudWatch Logs permite monitorar, armazenar e acessar os arquivos de log de instâncias do Amazon EC2, do CloudTrail e de outras fontes. O CloudWatch Logs pode monitorar informações nos arquivos de log e notificar você quando certos limites forem atingidos. É possível também arquivar seus dados de log em armazenamento resiliente. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch Logs.

Para obter recursos adicionais e as práticas recomendadas de segurança para o IAM, consulte Melhores práticas de segurança e casos de uso no AWS Identity and Access Management.

Tópicos

- Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail

- Rastreamento de tarefas privilegiadas no AWS CloudTrail

Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail

O IAM e o AWS STS são integrados ao AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário ou uma função do IAM. O CloudTrail captura todas as chamadas de API do IAM e do AWS STS como eventos, incluindo chamadas do console e de chamadas de API. Se você criar uma trilha, poderá habilitar a entrega contínua de eventos do CloudTrail para um bucket do Amazon S3. Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes no console do CloudTrail em Histórico de eventos. É possível usar o CloudTrail para obter informações sobre a solicitação que foi feita ao IAM ou ao AWS STS. Por exemplo, você pode visualizar o endereço IP de origem do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.

Tópicos

- Informações do IAM e do AWS STS no CloudTrail
- Registrar em log solicitações de API do IAM e do AWS STS
- Registrar em log solicitações de API em outros serviços da AWS
- Registrar em log eventos de login de usuário
- Registrar em log eventos de login de credenciais temporárias
- Exemplo de eventos de API do IAM no log do CloudTrail
- Exemplo de eventos de API do AWS STS no log do CloudTrail
- Exemplo de eventos de login no log do CloudTrail
- Comportamento da política de confiança de perfil do IAM

Informações do IAM e do AWS STS no CloudTrail

O CloudTrail é habilitado em sua Conta da AWS quando ela é criada. Quando ocorre uma atividade no IAM ou no AWS STS, essa atividade é registrada em um evento do CloudTrail junto com outros eventos de produtos da AWS em Event history (Histórico de eventos). Você pode exibir, pesquisar e baixar eventos recentes em sua Conta da AWS. Para saber mais, consulte Visualizar eventos com o histórico de eventos do CloudTrail.

Para obter um registro contínuo de eventos na sua Conta da AWS, incluindo eventos do IAM e do AWS STS, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as regiões da . A trilha registra logs de eventos de todas as Regiões na AWS divisória e entrega os arquivos do log para o bucket Amazon S3 especificado. Além disso, é possível configurar outros serviços da AWS para analisar mais ainda mais e agir com base nos dados de eventos coletados nos logs do CloudTrail. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- Serviços e integrações com suporte no CloudTrail
- Configurando Notificações Amazon SNS para CloudTrail
- Receber arquivos de log do CloudTrail de várias regiões e receber arquivos de log do CloudTrail de várias contas

Todas as ações do IAM e do AWS STS são registradas pelo CloudTrail e documentadas na Referência da API do IAM e na Referência da API do AWS Security Token Service.

Registrar em log solicitações de API do IAM e do AWS STS

O CloudTrail registra todas as solicitações de API autenticadas para o IAM e API do AWS STS. O CloudTrail também registra solicitações não autenticadas para as ações do AWS STS, `AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity` e registra informações fornecidas pelo provedor de identidade. No entanto, algumas solicitações não autenticadas do AWS STS podem não ser registradas em log porque não atendem à expectativa mínima de serem suficientemente válidas para serem consideradas uma solicitação legítima. O CloudTrail não registra em log solicitações do AWS STS negadas na conta de destino para sessões privilegiadas e suposições de perfil entre contas iniciadas por meio da API `AssumeRoot`.

É possível usar a informação registrada em log para mapear as chamadas feitas por uma entidade principal de usuário federado OIDC ou SAML com um perfil assumido de volta para o chamador federado externo de origem. No caso de `AssumeRole`, você pode mapear as chamadas de retorno feitas ao serviço AWS de origem ou à conta do usuário-fonte. A seção `userIdentity` dos dados JSON na entrada de log do CloudTrail contém as informações de que você precisa para mapear a solicitação AssumeRole* com uma entidade principal específica. Para obter mais informações, consulte o Elemento userIdentity do CloudTrail no Guia do usuário do AWS CloudTrail.

AWS CloudTrailOs logs do conterão informações de MFA quando o usuário do IAM fizer login com MFA. Se o usuário do IAM assumir um perfil do IAM, o CloudTrail também registrará em log

`mfaAuthenticated: true` nos atributos `sessionContext` para ações realizadas usando o perfil assumido. Porém, o registro em log do CloudTrail é separado do que o IAM exige quando as chamadas de API são feitas com as credenciais do perfil assumido. Para saber mais, consulte Elemento userIdentity do CloudTrail.

Por exemplo, as chamadas para o IAM `CreateUser`, `DeleteRole`, `ListGroups` e outras operações de API são todas registradas pelo CloudTrail.

Exemplos desse tipo de entrada de log são apresentados mais adiante neste tópico.

Registrar em log solicitações de API em outros serviços da AWS

Solicitações autenticadas a outras operações de API de produtos da AWS são registradas em log pelo CloudTrail, e esses regisros em log contêm informações sobre quem gerou a solicitação.

Por exemplo, suponha que você tenha feito uma solicitação para listar instâncias do Amazon EC2 ou para criar um grupo de implantação do AWS CodeDeploy. Há detalhes sobre a pessoa ou o serviço que fez a solicitação na entrada de log da solicitação. Essas informações ajudam a determinar se a solicitação foi feita pelo Usuário raiz da conta da AWS, por um usuário do IAM, por um perfil ou por outro serviço da AWS.

Para obter mais detalhes sobre as informações de identidade do usuário nos registros em log do CloudTrail, consulte o elemento userIdentity no Guia do usuário do AWS CloudTrail.

Registrar em log eventos de login de usuário

O CloudTrail registra em log eventos de login no Console de gerenciamento da AWS, em ferramentas de desenvolvimento local, como a AWS CLI e SDKs, nos fóruns de discussão da AWS e no AWS Marketplace. O CloudTrail registra em log tentativas de login bem-sucedidas e malsucedidas para usuários do IAM, entidades federadas SAML e OIDC e entidades principais de usuários federados do AWS STS.

Para visualizar exemplos de eventos do CloudTrail para logins de usuário-raiz bem-sucedidos e malsucedidos, consulte Exemplo de registros de eventos de usuários-raiz no Guia do usuário do AWS CloudTrail.

Como prática recomendada de segurança, a AWS não registra em log o texto do nome de usuário do IAM inserido quando a falha de login é causada por um nome de usuário incorreto. O texto do nome do usuário é mascarado pelo valor `HIDDEN_DUE_TO_SECURITY_REASONS`. Para obter um exemplo

disso, consulte Exemplo de evento de falha no login causado por nome de usuário incorreto mais adiante neste tópico. O texto do nome do usuário é obscurecido porque essas falhas talvez foram causadas por erros do usuário. Registrar em log esses erros pode expor informações confidenciais. Por exemplo:

- Você digita acidentalmente sua senha na caixa de nome do usuário.
- Você escolhe o link da página de login de uma Conta da AWS, mas digita o número de outra Conta da AWS.
- Você se esquece da conta na qual está fazendo login e acidentalmente digita o nome da sua conta de e-mail pessoal, o identificador de login de seu banco ou algum outro ID privado.

Registrar em log eventos de login de credenciais temporárias

Quando uma entidade principal solicita credenciais temporárias, o tipo de entidade principal determina como o CloudTrail registra o evento. Isso pode ser complicado quando um principal assume uma função em outra conta. Há várias chamadas de API para executar operações relacionadas a operações entre contas da função. Primeiro, o principal chama uma API do AWS STS para recuperar as credenciais temporárias. Essa operação está registrada na conta da chamada e na conta onde a operação do AWS STS é realizada. Depois disso, o principal usa a função para executar outras chamadas de API na conta da função assumida.

É possível usar a chave de condição `sts:SourceIdentity` na política de confiança do perfil para exigir que os usuários especifiquem uma identidade quando assumirem um perfil. Por exemplo, você pode exigir que os usuários do IAM especifiquem seu próprio nome de usuário como a identidade-fonte. Isso pode ajudar você a determinar qual usuário executou uma ação específica na AWS. Para obter mais informações, consulte `sts:SourceIdentity`. Você também pode usar `sts:RoleSessionName` para exigir que os usuários especifiquem um nome de sessão quando assumirem uma função. Isso pode ajudar você a diferenciar entre as sessões de função para uma função que é usada por diferentes entidades de segurança ao revisar os logs AWS CloudTrail.

A tabela a seguir mostra como o CloudTrail registra informações diferentes de identidade de usuário para cada uma das APIs AWS STS que geram credenciais temporárias.

Tipo da entidade principal	API DO STS	Identidade do usuário no log do CloudTrail da conta do chamador	Identidade do usuário no log do CloudTrail da conta da função assumida	Identidade do usuário no log do CloudTrail das chamadas de API subsequentes da função
Usuário raiz da conta da AWSCredenciais da	GetSessionToken	Identidade do usuário-raiz	A conta do proprietário do perfil é a mesma que a conta de chamada	Identidade do usuário-raiz
Usuário raiz da conta da AWSCredenciais da	AssumeRoot	Sessão do usuário-raiz	Número da conta e ID da entidade principal (se for um usuário)	Sessão do usuário-raiz
IAM user (Usuário do IAM)	GetSessionToken	Identidade do usuário do IAM	A conta do proprietário do perfil é a mesma que a conta de chamada	Identidade do usuário do IAM
IAM user (Usuário do IAM)	GetFederationToken	Identidade do usuário do IAM	A conta do proprietário do perfil é a mesma que a conta de chamada	Identidade do usuário do IAM
IAM user (Usuário do IAM)	AssumeRole	Identidade do usuário do IAM	Número da conta e ID principal (se for um usuário) ou serviço principal da AWS	Somente identidade do perfil (nenhum usuário)

Tipo da entidade principal	API DO STS	Identidade do usuário no log do CloudTrail da conta do chamador	Identidade do usuário no log do CloudTrail da conta da função assumida	Identidade do usuário no log do CloudTrail das chamadas de API subsequentes da função
Usuário autenticado externamente	AssumeRoleWithSAML	n/a	Identidade do usuário do SAML	Somente identidade do perfil (nenhum usuário)
Usuário autenticado externamente	AssumeRoleWithWebIdentity	n/a	Identidade do usuário da web/OIDC	Somente identidade do perfil (nenhum usuário)

O CloudTrail considera uma ação somente leitura se ela não tiver nenhum efeito mutante em um recurso. Ao registrar um evento somente leitura, o CloudTrail edita as informações do `responseElements` no log. Quando o CloudTrail registra um evento que não é somente leitura, o `responseElements` completo é mostrado na entrada do log. No entanto, para as APIs AWS STS `AssumeRole`, `AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity`, mesmo que estejam registradas como somente leitura, o CloudTrail incluirá o `responseElements` completo, com exceção de `secretAccessKey` no log dessas APIs.

A tabela a seguir mostra como o CloudTrail registra `responseElements` e informações `readOnly` para cada uma das APIs AWS STS que geram credenciais temporárias.

API DO STS	Informações dos elementos de resposta	Somente leitura
AssumeRole	Incluído	true
AssumeRoleWithSAML	Incluído	true
AssumeRoleWithWebIdentity	Incluído	true

API DO STS	Informações dos elementos de resposta	Somente leitura
AssumeRoot	Incluído	false
GetFederationToken	Incluído	false
GetSessionToken	Incluído	false

Exemplo de eventos de API do IAM no log do CloudTrail

Os arquivos de log do CloudTrail contêm eventos que são formatados com JSON. Um evento de API representa uma única solicitação de API e inclui informações sobre o principal, a ação solicitada, quaisquer parâmetros e a data e hora da ação.

Exemplo de evento de API do IAM no arquivo de log do CloudTrail

O exemplo a seguir mostra um registro em log do CloudTrail para uma solicitação pela ação `GetUserPolicy` do IAM.


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:Role-Session-Name",
        "arn": "arn:aws:sts::111122223333:assumed-role/Role-Name/Role-Session-Name",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-09-09T17:50:16Z",
                "mfaAuthenticated": "false"
            }
        }
```

```
    },
    "eventTime": "2024-09-09T17:51:44Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "GetUserPolicy",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.16.96 Python/2.7.8 Linux/10 botocore/1.12.86",
    "requestParameters": {
        "userName": "ExampleIAMUserName",
        "policyName": "ExamplePoliccyName"
    },
    "responseElements": null,
    "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
    "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    }
}
```

A partir dessas informações de evento, você pode determinar se a solicitação foi feita para obter uma política de usuário denominada `ReadOnlyAccess-JaneDoe-201407151307` para o usuário `JaneDoe`, como especificado no elemento `requestParameters`. Você também pode ver que a solicitação foi feita por uma usuária do IAM chamada `JaneDoe` em 15 de julho de 2014, às 21h40 (UTC). Nesse caso, a solicitação foi originada no Console de gerenciamento da AWS, como você pode observar no elemento `userAgent`.

Exemplo de eventos de API do AWS STS no log do CloudTrail

Os arquivos de log do CloudTrail contêm eventos que são formatados com JSON. Um evento de API representa uma única solicitação de API e inclui informações sobre o principal, a ação solicitada, quaisquer parâmetros e a data e hora da ação.

Exemplo de eventos de API do AWS STS entre contas em arquivos de log do CloudTrail

O usuário do IAM chamado `John` na conta 777788889999 chama a ação AssumeRole do AWS STS para assumir o perfil `EC2-dev` na conta 111122223333. O administrador da conta exige que os usuários definam uma identidade-fonte igual ao nome de usuário ao assumir a função. O usuário informa o valor de identidade-fonte `John`.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/John",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "John"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
    },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
```

```
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

O segundo exemplo mostra o registro em log do CloudTrail da conta da função assumida (111122223333) para a mesma solicitação.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "John-EC2-dev",
    "sourceIdentity": "John",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
```

```
    },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:John-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/John-EC2-dev"
      },
    "sourceIdentity": "John",
    "sessionTokenSize": 1428,
    "sessionTokenUtilization": 34
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

Exemplo de evento de API de encadeamento de funções do AWS STS no arquivo de log do CloudTrail

O exemplo a seguir mostra um registro em log do CloudTrail para uma solicitação feita por John Doe na conta 111111111111. John usou anteriormente seu usuário `John` para assumir a função `JohnRole1`. Para essa solicitação, ele usa as credenciais dessa função para assumir a função `JohnRole2`. Isso é conhecido como encadeamento de funções. A identidade-fonte que ele definiu quando assumiu a função `John1` persiste na solicitação para assumir `JohnRole2`. Se John tentar definir uma identidade-fonte diferente ao assumir a função, a solicitação será negada. John passa duas tags de sessão para a solicitação. Ele define essas duas tags como transitivas. A solicitação herda a tag `Department` como transitiva porque John a definiu como transitiva quando ele assumiu `JohnRole1`. Para obter mais informações sobre identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas. Para obter mais informações sobre chaves transitivas em cadeias de funções, consulte Encadeamento de funções com tags de sessão.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
```

```
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "John"
            },
            "sourceIdentity": "John"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "John",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
```

```
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
        "sourceIdentity": "John",
        "sessionTokenSize": 1652,
        "sessionTokenUtilization": 40
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```


Exemplo de evento de API do AWS STS de produto da AWS no arquivo de log do CloudTrail

O exemplo a seguir mostra um registro em log do CloudTrail para uma solicitação feita por um produto da AWS que chama outra API de serviço que usa as permissões de uma função de serviço. Ele mostra o registro em log do CloudTrail para a solicitação feita na conta 777788889999.


```
{
  "eventVersion": "1.04",
  "userIdentity": {
```

```
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "amzn-s3-demo-bucket"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```

Exemplo de evento de API SAML do AWS STS no arquivo de log do CloudTrail

O exemplo a seguir mostra um registro em log do CloudTrail para uma solicitação feita pela ação AssumeRoleWithSAML do AWS STS. A solicitação inclui os atributos SAML `CostCenter` e `Project` que são passados por meio da declaração do SAML como tags de sessão. Essas tags são definidas como transitivas para que persistam em cenários de encadeamento de funções. A solicitação inclui o parâmetro `DurationSeconds` opcional da API, que é representado como

`durationSeconds` no log do CloudTrail e é definido como `1800` segundos. A solicitação também inclui o atributo SAML `sourceIdentity`, que é passado na declaração SAML. Se alguém usar as credenciais de sessão de função resultantes para assumir outra função, essa identidade-fonte persistirá.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
```

```
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 34,
        "sessionTokenSize": 1428,
        "sessionTokenUtilization": 34,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```

Exemplo de evento da API AWS STS do OIDC no arquivo de log do CloudTrail

O exemplo seguinte mostra um registro de log do CloudTrail para uma solicitação feita pela ação AssumeRoleWithWebIdentity do AWS STS. A solicitação inclui os atributos `CostCenter` e `Project` que são passados por meio do token de provedor de identidades (IdP) OpenID Connect (OIDC) como tags de sessão. Essas etiquetas são definidas como transitivas para que persistam em caso de encadeamento de funções. A solicitação inclui o atributo `sourceIdentity` do token do provedor de identidade. Se alguém usar as credenciais de sessão de função resultantes para assumir outra função, essa identidade-fonte persistirá.

A entrada de log do CloudTrail também contém um campo `additionalEventData` com um atributo `identityProviderConnectionVerificationMethod`. Esse atributo indica o método da AWS usado para verificar a conexão com o provedor OIDC. O valor do atributo será `IAMTrustStore` ou `Thumbprint`. O valor `IAMTrustStore` indica que a AWS verificou com êxito a conexão com o IdP OIDC usando nossa biblioteca de autoridades de certificação raiz confiáveis (CAs). O valor `Thumbprint` indica que a AWS usou uma impressão digital de certificado definida na configuração do IdP para verificar o certificado do servidor do IdP OIDC.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>:<id of application>:<id of user>",
    "userName": "<id of user>",
    "identityProvider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
 provider>"
  },
  "eventTime": "2024-07-09T15:41:37Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/2.13.29 Python/3.11.6 Windows/10 exe/AMD64 prompt/off command/
sts.assume-role-with-web-identity",
  "requestParameters": {
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "<assigned role session name>",
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "principalTags": {
```

```
      "CostCenter": "24680",
      "Project": "Pegasus"
    },
    "transitiveTagKeys": [
      "CostCenter",
      "Project"
    ]
  },
  "responseElements": {
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "sessionToken": "<encoded session token blob>",
      "expiration": "Jul 9, 2024, 4:41:37 PM"
    },
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "assumedRoleUser": {
      "assumedRoleId": "AROA123456789EXAMPLE:<assigned role session name>",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/<assigned
role session name>"
    },
    "provider": "arn:aws:iam::444455556666:oidc-provider/<issuer url of OIDC
provider>",
    "audience": "<id of application>",
    "sessionTokenSize": 1536,
    "sessionTokenUtilization": 37
  },
  "additionalEventData": {
    "identityProviderConnectionVerificationMethod": "IAMTrustStore"
  },
  "requestID": "aEXAMPLE-0b26-40df-8973-c7012EXAMPLE",
  "eventID": "aEXAMPLE-ee29-4ac0-a0ed-3f5c5EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "444455556666",
      "type": "AWS::IAM::Role",
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "444455556666",
  "eventCategory": "Management",
```

```
  "tlsDetails": {
    "tlsVersion": "TLSv1.3",
    "cipherSuite": "TLS_AES_128_GCM_SHA256",
    "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
  }
}
```

Exemplo de evento de API AWS STS usando o endpoint global no arquivo de log do CloudTrail

Para solicitações ao endpoint global do AWS Security Token Service (AWS STS) (`https://sts.amazonaws.com`), o AWS STS inclui campos de log AWS CloudTrail adicionais: `endpointType` e `awsServingRegion`. Esses campos aparecem abaixo do elemento `additionalEventData` `RequestDetails` para registrar em log a Região da AWS responsável pelo atendimento e o tipo de endpoint que está sendo chamado. O campo `endpointType` pode ter um valor de `global` ou `regional` para indicar o tipo de endpoint global que atendeu à solicitação. Para obter mais informações sobre alterações em endpoints globais do AWS STS, consulte AWS STSRegiões e endpoints do.

> **Note**
>
> Os logs do AWS CloudTrail para solicitações feitas ao endpoint global do AWS STS serão enviados para a região Leste dos EUA (Norte da Virgínia). Os logs do CloudTrail para solicitações atendidas por endpoints regionais do AWS STS continuarão sendo registrados em suas respectivas regiões no CloudTrail.

O exemplo a seguir mostra uma entrada de log do CloudTrail para uma solicitação AWS STS feita ao endpoint global (`https://sts.amazonaws.com`) proveniente da região Europa (Estocolmo) - eu-north-1. O valor do campo `endpointType` de `global` indica que a solicitação AWS STS foi atendida pelo endpoint global na região Europa (Estocolmo).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
```

```
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:48Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:48 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "awsServingRegion": "eu-north-1",
            "endpointType": "global"
        }
```

```
    },
    "requestID": "EXAMPLE7-2497-457a-9586-f21feEXAMPLE",
    "eventID": "EXAMPLEc-3d26-4c3a-9c94-722a9EXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts-global.eu-north-1.amazonaws.com"
    }
}
```

Para comparação, o exemplo a seguir mostra uma entrada de log do CloudTrail para uma solicitação do AWS STS feita ao endpoint regional (`https://sts.us-west-2.amazonaws.com`) que foi atendida pelo endpoint regional na região Europa (Estocolmo) - eu-north-1. O valor do campo `endpointType` de `regional` indica que a solicitação AWS STS foi atendida pelo endpoint global na região Europa (Estocolmo).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA123456789EXAMPLE:developer",
        "arn": "arn:aws:sts::777788889999:assumed-role/Admin/developer",
        "accountId": "777788889999",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA123456789EXAMPLE",
                "arn": "arn:aws:iam::777788889999:role/Admin",
                "accountId": "777788889999",
                "userName": "Admin"
```

```
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2025-02-12T21:44:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2025-02-12T22:16:30Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "eu-north-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.15.33 Python/3.11.8 Linux/5.10.233-204.894.amzn2int.x86_64
 exe/x86_64.amzn.2 prompt/off command/sts.assume-role",
    "requestParameters": {
        "roleArn": "arn:aws:iam::777788889999:role/test-role",
        "roleSessionName": "test-global-assume-role"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Feb 12, 2025, 11:16:30 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROA987654321EXAMPLE:test-global-assume-role",
            "arn": "arn:aws:sts::777788889999:assumed-role/test-role/test-global-
assume-role"
        },
        "sessionTokenSize": 1096,
        "sessionTokenUtilization": 26
    },
    "additionalEventData": {
        "RequestDetails": {
            "endpointType": "regional",
            "awsServingRegion": "eu-north-1"
        }
    },
    "requestID": "EXAMPLEd-2116-4cd7-bd72-9f72fEXAMPLE",
    "eventID": "EXAMPLEd-219a-48ed-bc54-00e3cEXAMPLE",
    "readOnly": true,
    "resources": [
        {
```

```
            "accountId": "777788889999",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::777788889999:role/test-role"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "sts.eu-north-1.amazonaws.com"
    }
}
```

Exemplo de eventos de login no log do CloudTrail

Os arquivos de log do CloudTrail contêm eventos que são formatados com JSON. Um evento de login representa uma única solicitação de login e inclui informações sobre o principal de login, a região e a data e hora da ação.

Exemplo de evento de login bem-sucedido no arquivo de log do CloudTrail

O exemplo a seguir mostra um registro em log do CloudTrail para um evento de login bem-sucedido.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/John",
    "accountId": "111122223333",
    "userName": "John"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
```

```
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

O exemplo a seguir mostra uma entrada de log do CloudTrail para uma solicitação de código de autorização com êxito.

```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:thesjain-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/thesjain-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-17T22:50:14Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-17T22:51:32Z",
"eventSource": "signin.amazonaws.com",
"eventName": "AuthorizeOAuth2Access",
"awsRegion": "us-east-1",
"sourceIPAddress": "52.94.133.136",
"userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/142.0.0.0 Safari/537.36",
```

```
"requestParameters": {
"scope": "openid",
"redirect_uri": "http://127.0.0.1:53037/oauth/callback",
"code_challenge_method": "SHA-256",
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "e2854c76-1cba-4360-9fd1-5037b591466b",
"eventID": "59e1720d-3deb-44ff-933d-6828be2a860a",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```


O exemplo a seguir mostra uma entrada de log do CloudTrail para uma solicitação de criação de token OAuth2 com êxito.


```
{
"eventVersion": "1.11",
"userIdentity": {
"type": "AssumedRole",
"principalId": "AROATJHQDX737YZP72NTF:jacobjoj-Isengard",
"arn": "arn:aws:sts::225989345271:assumed-role/Admin/jacobjoj-Isengard",
"accountId": "225989345271",
"sessionContext": {
"sessionIssuer": {
"type": "Role",
"principalId": "AROATJHQDX737YZP72NTF",
"arn": "arn:aws:iam::225989345271:role/Admin",
```

```
"accountId": "225989345271",
"userName": "Admin"
},
"attributes": {
"creationDate": "2025-11-18T20:38:10Z",
"mfaAuthenticated": "false"
}
}
},
"eventTime": "2025-11-18T20:38:44Z",
"eventSource": "signin.amazonaws.com",
"eventName": "CreateOAuth2Token",
"awsRegion": "us-east-1",
"sourceIPAddress": "15.248.6.6",
"userAgent": "aws-cli/2.32.0 md/awscrt#0.28.4 ua/2.1 os/macos#24.6.0 md/arch#arm64
 lang/python#3.13.9 md/pyimpl#CPython m/b,AA,Z,E cfg/retry-mode#standard md/
installer#exe sid/35033f4ca1bd md/prompt#off md/command#login",
"requestParameters": {
"client_id": "arn:aws:signin:::devtools/same-device"
},
"responseElements": null,
"additionalEventData": {
"success": "true",
"x-amzn-vpce-id": ""
},
"requestID": "94562943-c85b-4dc1-bf72-43b0fd42d6de",
"eventID": "0b338fac-6a10-4740-b34d-1bb6923e799e",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "225989345271",
"eventCategory": "Management",
"tlsDetails": {
"tlsVersion": "TLSv1.3",
"cipherSuite": "TLS_AES_128_GCM_SHA256",
"clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
}
}
```


Para obter mais detalhes sobre as informações contidas nos arquivos de log do CloudTrail, consulte Referência de evento do CloudTrail no Guia do usuário do AWS CloudTrail.

Exemplo de evento de falha no login no arquivo de log do CloudTrail

O exemplo a seguir mostra um registro em log do CloudTrail para um evento de falha no login.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

Com base nessas informações, é possível determinar que a tentativa de login foi feita por uma usuária do IAM chamada `JaneDoe`, conforme mostrado no elemento `userIdentity`. Você também pode ver que houve falha nas tentativas de login, como mostrado no elemento `responseElements`. É possível ver que `JaneDoe` tentou fazer login no console do Amazon SNS às 17h35 (UTC) no dia 8 de julho de 2014.

Exemplo de evento de falha no login causado por nome de usuário incorreto

O exemplo a seguir mostra um registro em log do CloudTrail para um evento de login malsucedido causado pela digitação incorreta do nome do usuário. A AWS mascara o texto `userName` com

HIDDEN_DUE_TO_SECURITY_REASONS para ajudar a impedir a exposição de informações potencialmente confidenciais.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```


Comportamento da política de confiança de perfil do IAM

Em 21 de setembro de 2022, a AWS fez alterações no comportamento da política de confiança de perfil do IAM para exigir permissões explícitas em uma política de segurança de perfil quando um perfil assume a si mesmo. Os perfis do IAM na lista de permissões de comportamento legadas têm um campo additionalEventData em explicitTrustGrant para eventos de `AssumeRole`. O valor de `explicitTrustGrant` é falso quando um perfil na lista de permissões legadas assume a si mesmo usando o comportamento legado. Quando um perfil na lista de permissões legadas assume

a si mesmo, mas o comportamento da política de confiança de perfil foi atualizado para permitir explicitamente que o perfil assuma a si mesmo, o valor de `explicitTrustGrant` é verdadeiro.

Apenas um número muito pequeno de perfis do IAM está na lista de permissões para o comportamento legado, e esse campo só aparece nos logs do CloudTrail para esses perfis quando eles assumem a si mesmos. Na maioria dos casos, não é necessário que um perfil do IAM assuma a si mesmo. A AWS recomenda atualizar os processos, códigos ou configurações para remover esse comportamento ou atualizar as políticas de confiança de perfil para permitir explicitamente esse comportamento. Para obter mais informações, consulte Announcing an update to IAM role trust policy behavior.

Rastreamento de tarefas privilegiadas no AWS CloudTrail

A conta de gerenciamento do AWS Organizations ou uma conta de administrador delegado do IAM pode realizar algumas tarefas de usuário-raiz em contas-membro usando acesso raiz de curto prazo. Sessões privilegiadas de curto prazo fornecem credenciais temporárias que podem ser usadas para realizar ações privilegiadas em uma conta-membro em sua organização. É possível usar as etapas a seguir para identificar as ações realizadas pela conta de gerenciamento ou por um administrador delegado durante a sessão de `sts:AssumeRoot`.

> **Note**
>
> Não há suporte para o endpoint global em `sts:AssumeRoot`. O CloudTrail registra eventos de `ConsoleLogin` na região especificada para o endpoint.

Para rastrear ações realizadas por uma sessão privilegiada nos logs do CloudTrail

1. Encontre o evento `AssumeRoot` nos seus logs do CloudTrail. Esse evento é gerado quando sua conta de gerenciamento ou o administrador delegado do IAM obtém um conjunto de credenciais de curto prazo de `sts:AssumeRoot`.

   No exemplo a seguir, o evento CloudTrail para AssumeRoot é registrado em log no campo `eventName`.

   ```
   {
       "eventVersion": "1.08",
       "userIdentity": {
           "type": "AssumedRole",
           "principalId": "AIDACKCEVSQ6C2EXAMPLE:JohnRole1",
   ```

```
        "arn": "arn:aws:sts::111111111111:assumed-role/John/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/John",
                "accountId": "111111111111",
                "userName": "Admin2"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-10-25T20:45:28Z",
                "mfaAuthenticated": "false"
            },
            "assumedRoot": "true"
        }
    },
    "eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
        }
    }
}
```


Para obter as etapas para acessar seus logs do CloudTrail, consulte Obtenção e visualização de seus arquivos de log do CloudTrail no Guia do usuário do AWS CloudTrail.

2. No log de eventos do CloudTrail, localize o `targetPrincipal` o que especifica as ações da conta-membro que foram realizadas e o `accessKeyId` que é exclusivo da sessão de `AssumeRoot`.

   No exemplo a seguir, o `targetPrincipal` é 222222222222 e o `accessKeyId` é ASIAIOSFODNN7EXAMPLE.

```
"eventTime": "2024-10-25T20:52:11Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoot",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "targetPrincipal": "222222222222",
        "taskPolicyArn": {
            "arn": "arn:aws:iam::aws:policy/root-task/S3UnlockBucketPolicy"
        }
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
            "expiration": "Oct 25, 2024, 9:07:11 PM"
    }
}
```

3. Nos logs do CloudTrail para a entidade principal alvo, pesquise o ID da chave de acesso que corresponde ao valor `accessKeyId` do evento `AssumeRoot`. Use os valores do campo `eventName` para determinar as tarefas privilegiadas realizadas durante a sessão de `AssumeRoot`. Pode haver várias tarefas privilegiadas executadas em uma única sessão. A duração máxima de sessão para `AssumeRoot` é de 900 segundos (15 minutos).

   No exemplo a seguir, a conta de gerenciamento ou o administrador delegado excluiu a política baseada em recursos para um bucket do Amazon S3.

```
{
    "eventVersion": "1.10",
    "userIdentity": {
        "type": "Root",
        "principalId": "222222222222",
        "arn": "arn:aws:iam::222222222222:root",
        "accountId": "222222222222",
```

```
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "creationDate": "2024-10-25T20:52:11Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-10-25T20:53:47Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "DeleteBucketPolicy",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "requestParameters": {
        "bucketName": "resource-policy-John",
        "Host": "resource-policy-John.s3.amazonaws.com",
        "policy": ""
    },
    "responseElements": null,
    "requestID": "1234567890abcdef0",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "readOnly": false,
    "resources": [
        {
            "accountId": "222222222222",
            "type": "AWS::S3::Bucket",
            "ARN": "arn:aws:s3:::resource-policy-John"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "222222222222",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "resource-policy-John.s3.amazonaws.com"
    }
}
```

Validação de conformidade do AWS Identity and Access Management

Auditores externos avaliam a segurança e a conformidade do AWS Identity and Access Management (IAM) como parte de vários programas de conformidade da AWS. Isso inclui SOC, PCI, FedRAMP, ISO e outros.

Nossa nova experiência de cadastro na AWS não foi projetada para workloads regulamentadas. Se você estiver usando nossa nova experiência de login da AWS, mas quiser usar a AWS para workloads regulamentadas, você pode fazer login na AWS (avançado) ou ativar os atributos avançados para o ambiente da AWS.

Para saber se um AWS service (Serviço da AWS) está no escopo de programas específicos de conformidade, consulte Serviços da AWS no escopo por programa de conformidade e escolha o programa de conformidade de seu interesse. Para obter informações gerais, consulte Programas de Conformidade da AWS.

É possível baixar relatórios de auditoria de terceiros usando o AWS Artifact. Para saber mais, consulte Baixar relatórios no AWS Artifact.

Sua responsabilidade de conformidade ao usar o Serviços da AWS é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da sua empresa e pelos regulamentos e leis aplicáveis. Para ter mais informações sobre sua responsabilidade de conformidade ao usar Serviços da AWS, consulte Documentação de segurança da AWS.

Resiliência no AWS Identity and Access Management

A infraestrutura global da AWS é criada com base em regiões da AWS e zonas de disponibilidade. As regiões da AWS As regiões têm várias zonas de disponibilidade fisicamente separadas e isoladas, que são conectadas com redes de baixa latência, alto throughput e alta redundância. Para obter mais informações sobre regiões e zonas de disponibilidade da AWS, consulte Infraestrutura global da AWS.

O AWS Identity and Access Management (IAM) e o AWS Security Token Service (AWS STS) são serviços autossustentáveis baseados em região que estão disponíveis globalmente.

O IAM é um AWS service (Serviço da AWS) essencial. Toda operação realizada na AWS deve ser autenticada e autorizada pelo IAM. O IAM verifica cada solicitação em relação às identidades e políticas armazenadas no IAM para determinar se a solicitação é permitida ou negada. O IAM foi

projetado com um ambiente de gerenciamento e um plano de dados separados para que o serviço faça autenticações mesmo durante falhas inesperadas. Os recursos do IAM usados em autorizações, como funções e políticas, são armazenados no ambiente de gerenciamento. Os clientes do IAM podem alterar a configuração desses recursos usando as operações do IAM, como `DeletePolicy` e `AttachRolePolicy`. Essas solicitações de alteração de configuração vão para o ambiente de gerenciamento. Há um único ambiente de gerenciamento do IAM para todas as Regiões da AWS comerciais e ele fica localizado na região Leste dos EUA (Norte da Virgínia). O sistema do IAM propaga as alterações de configuração para os planos de dados do IAM de toda Região da AWS habilitada. O plano de dados do IAM é essencialmente uma réplica somente leitura dos dados de configuração do ambiente de gerenciamento do IAM. Cada Região da AWS tem uma instância completamente independente do plano de dados do IAM que realiza a autenticação e a autorização das solicitações da mesma região. Em cada região, o plano de dados do IAM é distribuído por pelo menos três zonas de disponibilidade e tem capacidade suficiente para suportar a perda de uma zona de disponibilidade sem qualquer prejuízo para o cliente. Tanto o controle do IAM quanto os planos de dados foram criados para não ter nenhuma paralização planejada, sendo todas as atualizações de software e operações de escala realizadas de maneira invisível para os clientes.

Nas regiões que são habilitadas por padrão, as solicitações para o endpoint global do AWS STS são atendidas automaticamente na mesma região de origem da solicitação. Nas regiões de adesão, as solicitações para o endpoint global do AWS STS são atendidas por uma única Região da AWS, Leste dos EUA (Norte da Virgínia). Porém, você pode usar um endpoint regional do AWS STS para reduzir latência ou fornecer redundância adicional para as aplicações. Para saber mais, consulte Gerenciar o AWS STS em uma Região da AWS.

Certos eventos podem interromper a comunicação entre Regiões da AWS pela rede. Porém, mesmo quando você não pode se comunicar com o endpoint global do IAM, o AWS STS ainda pode autenticar as entidades principais do IAM e o IAM pode autorizar suas solicitações. Os detalhes específicos de um evento que interrompe a comunicação determinarão sua capacidade de acessar os serviços da AWS. Na maioria das situações, você pode continuar a usar credenciais do IAM no ambiente da AWS. As condições a seguir podem se aplicar a um evento que interrompe a comunicação.

Chaves de acesso de usuários do IAM

> Você pode fazer autenticações indefinidamente em uma região com chaves de acesso de usuários do IAM de longo prazo. Quando você usa a AWS Command Line Interface e as APIs, você pode fornecer chaves de acesso da AWS para que a AWS possa verificar sua identidade em solicitações programáticas.

> **Important**
>
> Como prática recomendada, sugerimos que os usuários façam login com credenciais temporárias em vez de chaves de acesso de longo prazo.

Credenciais temporárias

Você pode solicitar novas credenciais temporárias com o endpoint de serviço regional do AWS STS para, pelo menos, 24 horas. As operações de API a seguir geram credenciais temporárias.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

Entidades principais e permissões

- Talvez você não possa adicionar, modificar ou remover permissões ou entidades principais no IAM.
- Suas credenciais podem não refletir as alterações nas permissões que você aplicou recentemente no IAM. Para obter mais informações, consulte As alterações que eu faço nem sempre ficam imediatamente visíveis.

Console de gerenciamento da AWS

- Talvez você possa usar um endpoint de login regional para entrar no Console de gerenciamento da AWS como um usuário do IAM. Os endpoints de login regionais têm o seguinte formato de URL.

  `https://{Account ID}.signin.aws.amazon.com/console?region={Region}`

  Exemplo: https://111122223333.signin.aws.amazon.com/console?region=us-west-2
- Talvez você não possa realizar a autenticação multifator (MFA) Universal 2nd Factor (U2F).

Práticas recomendadas para a resiliência do IAM

A AWS incorporou resiliência nas zonas de disponibilidade e Regiões da AWS. Quando você observa as práticas recomendadas do IAM a seguir nos sistemas que interagem com seu ambiente, aproveita essa resiliência.

1. Use um endpoint de serviço regional da AWS STS em vez do endpoint global padrão.
2. Analise a configuração do ambiente em busca de recursos vitais que rotineiramente criam ou modificam recursos do IAM e prepare uma solução de fallback que use os recursos existentes do IAM.

Segurança da infraestrutura no AWS Identity and Access Management

Como serviço gerenciado, o AWS Identity and Access Management é protegido pela segurança de rede global da AWS. Para saber mais sobre serviços de segurança da AWS e como a AWS protege a infraestrutura, consulte Segurança na Nuvem AWS. Para projetar seu ambiente da AWS usando as práticas recomendadas de segurança de infraestrutura, consulte Proteção de infraestrutura em Pilar segurança: AWS Well-Architected Framework.

Você usa chamadas de API publicadas pela AWS para acessar o IAM pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

O IAM pode ser acessado de forma programática usando a API HTTPS do IAM, que permite emitir solicitações HTTPS diretamente ao serviço. A API de consulta retorna informações confidenciais, incluindo credenciais de segurança. Portanto, é necessário usar HTTPS com todas as solicitações de API. Quando você usa a API HTTPS, deve incluir código para assinar digitalmente solicitações usando suas credenciais.

Você pode chamar essas operações de API de qualquer local da rede, mas o IAM oferece suporte a políticas de acesso baseadas em recurso, que podem incluir restrições com base no endereço IP da

fonte. Você também pode usar políticas do IAM para controlar o acesso de endpoints específicos da Amazon Virtual Private Cloud (Amazon VPC) ou VPCs específicas. Efetivamente, isso isola o acesso à rede para um determinado recurso do IAM somente da VPC específica dentro da rede da AWS.

Análise de vulnerabilidade e configuração no AWS Identity and Access Management

AWSA se encarrega das tarefas básicas de segurança, como aplicação de patches a bancos de dados e sistemas operacionais (SOs) convidados, configuração de firewalls e recuperação de desastres. Esses procedimentos foram revisados e certificados por terceiros certificados. Para obter mais detalhes, consulte os recursos a seguir:

- Modelo de responsabilidade compartilhada
- Amazon Web Services: visão geral do processo de segurança (whitepaper)

Os recursos a seguir também abordam a análise de configuração e vulnerabilidade no AWS Identity and Access Management (IAM):

- Validação de conformidade do AWS Identity and Access Management
- Melhores práticas de segurança e casos de uso no AWS Identity and Access Management

Políticas da AWS gerenciadas para AWS Identity and Access Management, IAM Access Analyzer e gerenciador de acesso à conta

Uma política gerenciada pela AWS é uma política autônoma criada e administrada pela AWS. As políticas gerenciadas pela AWS são criadas para fornecer permissões a vários casos de uso comuns e permitir a atribuição de permissões a usuários, grupos e perfis.

Lembre-se de que as políticas gerenciadas pela AWS podem não conceder permissões de privilégio mínimo para casos de uso específicos, por estarem disponíveis para uso por todos os clientes da AWS. Recomendamos que você reduza ainda mais as permissões definindo as políticas gerenciadas pelo cliente que são específicas para seus casos de uso.

Não é possível alterar as permissões definidas em políticas gerenciadas pela AWS. Se a AWS atualiza as permissões definidas em um política gerenciada por AWS, a atualização afeta todas as identidades de entidades principais (usuários, grupos e perfis) às quais a política estiver vinculada. É provável que a AWS atualize uma política gerenciada por AWS quando um novo AWS service (Serviço da AWS) for lançado, ou novas operações de API forem disponibilizadas para os serviços existentes.

Para saber mais, consulte AWSPolíticas gerenciadas pela no Guia do usuário do IAM.

IAMReadOnlyAccess

Use a política gerenciada `IAMReadOnlyAccess` para permitir acesso do tipo somente leitura a recursos do IAM. Essa política concede permissão para obter e listar todos os recursos do IAM. Ela permite visualizar detalhes e relatórios de atividades para usuários, grupos, funções, políticas, provedores de identidade e dispositivos com MFA. A política não inclui a capacidade de criar ou excluir recursos, ou de acessar os recursos do IAM Access Analyzer. Consulte a política para obter a lista completa de serviços e ações suportados por essa política.

IAMUserChangePassword

Use a política gerenciada `IAMUserChangePassword` para permitir que usuários do IAM alterem suas senhas.

Você define suas Configurações de conta do IAM e a Política de senhas para permitir que os usuários do IAM alterem suas senhas da conta do IAM. Ao permitir essa ação, o IAM associa a política individualmente a cada usuário.

Para visualizar a política em JSON, consulte IAMUserChangePassword no Guia de referência de políticas gerenciadas pela AWS.

IAMAccessAnalyzerFullAccess

Usar a política gerenciada pela AWS `IAMAccessAnalyzerFullAccess` para permitir que seus administradores acessem o IAM Access Analyzer.

Agrupamentos de permissões

Esta política é agrupada em declarações com base no conjunto de permissões fornecidas.

- IAM Access Analyzer: concede permissões administrativas totais para todos os recursos no IAM Access Analyzer.
- Criar função vinculada ao serviço: permite que o administrador crie uma função vinculada ao serviço, que permite que o IAM Access Analyzer analise recursos em outros serviços em seu nome. Essa permissão permite criar a função vinculada ao serviço somente para uso pelo IAM Access Analyzer.
- AWS Organizations – permite que os administradores usem o IAM Access Analyzer para uma organização no AWS Organizations. Depois de habilitar o acesso confiável para o IAM Access Analyzer no AWS Organizations, os membros da conta de gerenciamento podem visualizar as descobertas em toda a organização.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
```

```
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

IAMAccessAnalyzerReadOnlyAccess

Use a política gerenciada pela AWS `IAMAccessAnalyzerReadOnlyAccess` para permitir acesso somente leitura ao IAM Access Analyzer.

Para permitir acesso somente leitura ao IAM Access Analyzer para o AWS Organizations, crie uma política gerenciada pelo cliente que permita as ações Describe (Descrever) e List (Listar) da política IAMAccessAnalyzerFullAccess gerenciada pela AWS.

Permissões no nível do serviço

Esta política fornece acesso somente leitura ao IAM Access Analyzer. Nenhuma outra permissão de serviço está incluída nesta política.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolicy

Você não pode anexar AccessAnalyzerServiceRolePolicy às entidades do IAM. Esta política está anexada a uma função vinculada ao serviço que permite ao IAM Access Analyzer executar ações em seu nome. Para obter mais informações, consulte Usar funções vinculadas ao serviço do AWS Identity and Access Management and Access Analyzer.

Agrupamentos de permissões

Esta política permite acesso ao IAM Access Analyzer para analisar metadados de recursos de vários Serviços da AWS.

- Amazon DynamoDB: concede permissões para visualizar streams e tabelas do DynamoDB.
- Amazon Elastic Compute Cloud: concede permissões para descrever endereços IP, snapshots e VPCs.
- Amazon Elastic Container Registry: concede permissões para descrever repositórios de imagens, recuperar configurações de contas e recuperar políticas de repositórios e registros.
- Amazon Elastic File System: concede permissões para visualizar a descrição de um sistema de arquivos do Amazon EFS e visualizar a política em nível de recurso para um sistema de arquivos do Amazon EFS.
- AWS Identity and Access Management – concede permissões para recuperar informações sobre uma função específica e listar os perfis do IAM que têm um prefixo de caminho específico. Aceita permissões para recuperar informações sobre usuários, grupos do IAM, perfis de login, chaves de acesso e dados acessados mais recentemente do serviço.
- AWS Key Management Service – concede permissões para visualizar informações detalhadas sobre uma chave do KMS e suas principais políticas e concessões.
- AWS Lambda – concede permissões para visualizar informações sobre aliases do Lambda, funções, camadas e aliases.
- AWS Organizations: concede permissões para o AWS Organizations e permite a criação de um analisador na organização da AWS como a zona de confiança.

- Amazon Relational Database Service: concede permissões para visualizar informações detalhadas sobre snapshots de banco de dados do Amazon RDS e snapshots de cluster de banco de dados do Amazon RDS.
- Amazon Simple Storage Service: concede permissões para visualizar informações detalhadas sobre Pontos de Acesso Amazon S3 e buckets, pontos de acesso do bucket de diretório do Amazon S3 e buckets de diretório.
- AWS Secrets Manager – concede permissões para visualizar informações detalhadas sobre segredos e políticas de recursos anexadas a segredos.
- Amazon Simple Notification Service: concede permissões para visualizar informações detalhadas sobre um tópico.
- Amazon Simple Notification Service: concede permissões para visualizar informações detalhadas sobre filas específicas.

Para visualizar a política de JSON, consulte AccessAnalyzerServiceRolePolicy no Guia de referência de políticas gerenciadas da AWS.

IAMAuditRootUserCredentials

Use a política `IAMAuditRootUserCredentials` gerenciada pela AWS para reduzir o escopo das permissões ao realizar uma tarefa privilegiada em uma conta-membro do AWS Organizations para auditar o status da credencial de usuário-raiz de uma conta-membro. Você pode listar ou obter informações individuais de credenciais de usuário-raiz, como:

- Se há uma senha de usuário-raiz
- Se o usuário-raiz tem uma chave de acesso e quando ela foi usada pela última vez
- Se o usuário-raiz tem certificados de assinatura associados
- Os dispositivos de MFA associados ao usuário-raiz
- Lista do status consolidado das credenciais do usuário-raiz

Não é possível anexar a `IAMAuditRootUserCredentials` às entidades do IAM. Essa política é anexada a AssumeRoot para realizar tarefas privilegiadas em uma conta de membro em sua organização. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices",
            "iam:GetAccountSummary",
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyAuditingCredentialsOnNonRootUserResource",
         "Action": [
            "iam:ListAccessKeys",
            "iam:ListSigningCertificates",
            "iam:GetLoginProfile",
            "iam:ListMFADevices" ,
            "iam:GetUser",
            "iam:GetAccessKeyLastUsed"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

Agrupamentos de permissões

Esta política é agrupada em declarações com base no conjunto de permissões fornecidas.

- DenyAllOtherActionsOnAnyResource: nega o acesso às credenciais para todos os recursos.

- DenyAuditingCredentialsOnNonRootUserResource: nega o acesso às credenciais para todos os recursos não de usuário-raiz.

IAMCreateRootUserPassword

Use a política `IAMCreateRootUserPassword` gerenciada pela AWS para reduzir o escopo das permissões ao realizar uma tarefa privilegiada em uma conta-membro do AWS Organizations para permitir a recuperação de senha para uma conta-membro sem credenciais de usuário-raiz.

Não é possível anexar a `IAMCreateRootUserPassword` às entidades do IAM. Essa política é anexada a AssumeRoot para realizar tarefas privilegiadas em uma conta de membro em sua organização. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyCreatingPasswordOnNonRootUserResource",
         "Action": [
            "iam:CreateLoginProfile",
            "iam:GetLoginProfile"
         ],
         "Effect": "Deny",
         "NotResource": "arn:aws:iam::*:root"
      }
   ]
}
```

Agrupamentos de permissões

Esta política é agrupada em declarações com base no conjunto de permissões fornecidas.

- DenyAllOtherActionsOnAnyResource: nega acesso para obter ou criar uma senha para todos os recursos.
- DenyCreatingPasswordOnNonRootUserResource: nega acesso para obter ou criar uma senha para todos os recursos não de usuário-raiz.

IAMDeleteRootUserCredentials

Use a política `IAMDeleteRootUserCredentials` gerenciada pela AWS para reduzir o escopo das permissões ao realizar uma tarefa privilegiada em uma conta-membro do AWS Organizations para remover as credenciais de usuário-raiz, incluindo senha, chaves de acesso e assinatura de certificados e desativar a MFA. Permissões adicionais são necessárias para essa ação privilegiada, para que você possa visualizar as informações da última credencial usada, verificar as últimas informações usadas para o usuário-raiz da conta-membro e listar as permissões para que todas as credenciais de usuário-raiz sejam excluídas.

Não é possível anexar a `IAMDeleteRootUserCredentials` às entidades do IAM. Essa política é anexada a AssumeRoot para realizar tarefas privilegiadas em uma conta de membro em sua organização. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "DenyAllOtherActionsOnAnyResource",
   "Effect": "Deny",
   "NotAction": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
```

```
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "Resource": "*"
  },
  {
   "Sid": "DenyDeletingRootUserCredentialsOnNonRootUserResource",
   "Effect": "Deny",
   "Action": [
    "iam:DeleteAccessKey",
    "iam:DeleteSigningCertificate",
    "iam:DeleteLoginProfile",
    "iam:DeactivateMFADevice",
    "iam:ListAccessKeys",
    "iam:ListSigningCertificates",
    "iam:GetLoginProfile",
    "iam:ListMFADevices",
    "iam:GetUser",
    "iam:GetAccessKeyLastUsed"
   ],

   "NotResource": "arn:aws:iam::*:root"
  }
 ]
}
```

Agrupamentos de permissões

Esta política é agrupada em declarações com base no conjunto de permissões fornecidas.

- DenyAllOtherActionsOnAnyResource: nega o acesso para obter ou excluir credenciais para todos os recursos.
- DenyDeletingRootUserCredentialsOnNonRootUserResource: nega o acesso para obter ou excluir as credenciais para todos os recursos não de usuário-raiz.

S3UnlockBucketPolicy

Use a política `S3UnlockBucketPolicy` gerenciada pela AWS para reduzir o escopo das permissões ao realizar uma tarefa privilegiada em uma conta-membro do AWS Organizations para remover uma política de bucket mal configurada que impeça que todas as entidades principais acessem um bucket do Amazon S3.

Não é possível anexar a `S3UnlockBucketPolicy` às entidades do IAM. Essa política é anexada a AssumeRoot para realizar tarefas privilegiadas em uma conta de membro em sua organização. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "DenyAllOtherActionsOnAnyResource",
         "NotAction": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*"
      },
      {
         "Sid": "DenyManagingBucketPolicyForNonRootCallers",
         "Action": [
            "s3:DeleteBucketPolicy",
            "s3:PutBucketPolicy",
            "s3:GetBucketPolicy",
            "s3:ListAllMyBuckets"
         ],
         "Effect": "Deny",
         "Resource": "*",
         "Condition" : {
            "ArnNotLike" : {
               "aws:PrincipalArn" : "arn:aws:iam::*:root"
            }
         }
```

```
        }
    ]
}
```

Agrupamentos de permissões

Esta política é agrupada em declarações com base no conjunto de permissões fornecidas.

- DenyAllOtherActionsOnAnyResource: nega o acesso a políticas de bucket para todos os recursos.
- DenyManagingBucketPolicyForNonRootCallers: nega o acesso a políticas de bucket para todos os recursos não de usuário-raiz.

SQSUnlockQueuePolicy

Use a política `SQSUnlockQueuePolicy` gerenciada pela AWS para reduzir o escopo das permissões ao realizar uma tarefa privilegiada em uma conta-membro do AWS Organizations para excluir uma política baseada em recurso do Amazon Simple Queue Service que negue a todas as entidades principais o acesso a uma fila do Amazon SQS.

Não é possível anexar a `SQSUnlockQueuePolicy` às entidades do IAM. Essa política é anexada a AssumeRoot para realizar tarefas privilegiadas em uma conta de membro em sua organização. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

Agrupamentos de permissões

Esta política é agrupada em declarações com base no conjunto de permissões fornecidas.

- DenyAllOtherActionsOnAnyResource: nega o acesso a ações do Amazon SQS para todos os recursos.
- DenyGettingQueueAttributesOnNonOwnQueue: nega acesso aos atributos de fila do Amazon SQS para filas pertencentes a outra conta.
- DenyActionsForNonRootUser: nega o acesso a ações do Amazon SQS para todos os recursos não de usuário-raiz.

AccountAccessManagerServiceRolePolicy

Não é possível anexar a `AccountAccessManagerServiceRolePolicy` às entidades do IAM. Essa política é anexada a um perfil vinculado ao serviço que permite que o gerenciador de acesso à conta realize ações em seu nome. Para obter mais informações, consulte the section called "Segurança".

Permissões no nível do serviço

Essa política permite que o gerenciador de acesso à conta recupere informações sobre suas contas do Organizations para manter uma associação precisa à conta para gerenciamento de acesso. O gerenciador de acesso à conta usa essas permissões para detectar quando as contas saem da sua organização, verificar se o acesso a serviços confiáveis está habilitado e confirmar o status da conta antes de gerenciar os direitos.

- AWS Organizations — Concede permissões para listar contas na organização, verificar se o acesso a serviços confiáveis está habilitado e descrever o status da conta.

Para visualizar a política de JSON, consulte AccountAccessManagerServiceRolePolicy no Guia de referência de políticas gerenciadas da AWS.

Atualizações do IAM, do IAM Access Analyzer e do gerenciador de acesso à conta quanto às políticas gerenciadas da AWS

Veja detalhes sobre atualizações do IAM e políticas gerenciadas da AWS desde que esse serviço começou a monitorar essas alterações. Para obter alertas automáticos sobre alterações nesta página, assine o feed RSS nas páginas de históricos de documentos do IAM e do IAM Access Analyzer.

Alteração	Descrição	Data
AccountAccessManagerServiceRolePolicy — Política gerenciada adicionada	O gerenciador de acesso à conta adicionou uma política de perfil vinculada ao serviço que permite que o serviço	31 de julho de 2026

Alteração	Descrição	Data
	recupere informações sobre suas contas do Organizations.	
IAMUserChangePassword: permissões adicionadas	O IAM adicionou permissões à política `IAMUserChangePassword` para permitir que usuários especificados em um determinado caminho possam alterar a senha.	28 de maio de 2025
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou `iam:GetAccountAuthorizationDetails` para as permissões de nível de serviço do `AccessAnalyzerServiceRolePolicy` .	12 de maio de 2025
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O Analisador de Acesso do IAM adicionou suporte pontos de acesso dos buckets de diretório do Amazon S3 para as permissões de nível de serviço de `AccessAnalyzerServiceRolePolicy` .	31 de março de 2025
IAMDeleteRootUserCredentials: permissões removidas	O IAM removeu a permissão `iam:DeleteVirtualMFADevice` da política gerenciada.	7 de janeiro de 2025

Alteração	Descrição	Data
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte à permissão para recuperar informações sobre políticas de registros e configurações de conta do Amazon ECR para as permissões no nível de serviço de `AccessAnalyzerServiceRolePolicy` .	10 de dezembro de 2024
IAMAuditRootUserCredentials: política gerenciada adicionada	O IAM adicionou políticas gerenciadas para Gerenciamento central do acesso raiz às contas-membro para definir tarefas privilegiadas que podem ser realizadas nas contas-membro do AWS Organizations.	14 de novembro de 2024
IAMCreateRootUserPassword: política gerenciada adicionada	O IAM adicionou políticas gerenciadas para Gerenciamento central do acesso raiz às contas-membro para definir tarefas privilegiadas que podem ser realizadas nas contas-membro do AWS Organizations.	14 de novembro de 2024

Alteração	Descrição	Data
IAMDeleteRootUserCredentials: política gerenciada adicionada	O IAM adicionou políticas gerenciadas para Gerenciamento central do acesso raiz às contas-membro para definir tarefas privilegiadas que podem ser realizadas nas contas-membro do AWS Organizations.	14 de novembro de 2024
S3UnlockBucketPolicy: política gerenciada adicionada	O IAM adicionou políticas gerenciadas para Gerenciamento central do acesso raiz às contas-membro para definir tarefas privilegiadas que podem ser realizadas nas contas-membro do AWS Organizations.	14 de novembro de 2024
SQSUnlockQueuePolicy: política gerenciada adicionada	O IAM adicionou políticas gerenciadas para Gerenciamento central do acesso raiz às contas-membro para definir tarefas privilegiadas que podem ser realizadas nas contas-membro do AWS Organizations.	14 de novembro de 2024
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte aàpermissão para recuperar informações sobre tags de perfil e usuário do IAM para as permissões de nível de serviço de `AccessAnalyzerServiceRolePolicy` .	29 de outubro de 2024

Alteração	Descrição	Data
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte à permissão para recuperar informações sobre políticas de usuário e perfil do IAM para as permissões de nível de serviço de `AccessAnalyzerServiceRolePolicy` .	30 de maio de 2024
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte à permissão para recuperar o estado atual do bloco de acesso público para instantâneos do Amazon EC2 para as permissões de nível de serviço de `AccessAnalyzerServiceRolePolicy` .	23 de janeiro de 2024
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte a streams e tabelas do DynamoDB às permissões de nível de serviço do `AccessAnalyzerServiceRolePolicy` .	11 de janeiro de 2024
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte a buckets de diretório do Amazon S3 para as permissões de nível de serviço de `AccessAnalyzerServiceRolePolicy` .	1.º de dezembro de 2023

Alteração	Descrição	Data
IAMAccessAnalyzerReadOnlyAccess: permissões adicionadas	O IAM Access Analyzer adicionou permissões para permitir que você verifique se as atualizações em suas políticas concedem acesso adicional. Essa permissão é exigida pelo IAM Access Analyzer para executar verificações de política em suas políticas.	26 de novembro de 2023
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou ações IAM para as permissões de nível de serviço de `AccessAnalyzerServiceRolePolicy` para dar suporte às seguintes ações: • Listando entidades para uma política • Gerando detalhes do serviço acessados pela última vez • Listar informações de chave de acesso	26 de novembro de 2023

Alteração	Descrição	Data
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou suporte para os seguintes tipos de recursos para as permissões em nível de serviço de `AccessAnalyzerServiceRolePolicy` : • Snapshots de volume do Amazon EBS • Repositórios do Amazon ECR • Sistemas de arquivos do Amazon EFS • Snapshots de banco de dados do Amazon RDS • Snapshots de cluster de banco de dados do Amazon RDS • Tópicos do Amazon SNS	25 de outubro de 2022
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou a ação `lambda:GetFunctionUrlConfig` para as permissões de nível de serviço do `AccessAnalyzerServiceRolePolicy` .	6 de abril de 2022

Alteração	Descrição	Data
AccessAnalyzerServiceRolePolicy: permissões adicionadas	O IAM Access Analyzer adicionou novas ações do Amazon S3 para analisar metadados associados a pontos de acesso multirregiões.	2 de setembro de 2021
IAMAccessAnalyzerReadOnlyAccess: permissões adicionadas	O IAM Access Analyzer adicionou uma nova ação para conceder permissões `ValidatePolicy` para permitir que você use as verificações de política para validação. Essa permissão é exigida pelo IAM Access Analyzer para executar verificações de política em suas políticas.	16 de março de 2021
O IAM Access Analyzer começou a monitorar alterações	O IAM Access Analyzer começou a rastrear alterações para suas políticas gerenciadas pela AWS.	1.º de março de 2021

Recursos de segurança fora do IAM

Você usa o IAM para controlar o acesso a tarefas que são executadas usando o Console de gerenciamento da AWS, as ferramentas da linha de comando da AWS ou operações da API de serviço usando os SDKs da AWS. Alguns produtos da AWS têm outras maneiras de proteger seus recursos. A lista a seguir fornece alguns exemplos, embora não seja completa.

Amazon EC2

No Amazon Elastic Compute Cloud, você faz login em uma instância com um par de chaves (para instâncias do Linux) ou usando um nome de usuário e senha (para instâncias do Microsoft Windows).

Para obter mais informações, consulte a seguinte documentação do :

- Comece a usar instâncias de Linux do Amazon EC2 no Guia do usuário do Amazon EC2
- Comece a usar instâncias de Windows do Amazon EC2 no Guia do usuário do Amazon EC2

Amazon RDS

No Amazon Relational Database Service, você faz login no mecanismo de banco de dados com um nome de usuário e senha vinculados a esse banco de dados.

Para obter mais informações, consulte Conceitos básicos do Amazon RDS no Guia do usuário do Amazon RDS.

Amazon EC2 e Amazon RDS

No Amazon EC2 e no Amazon RDS, você pode usar grupos de segurança para controlar o tráfego para uma instância ou banco de dados.

Para obter mais informações, consulte a seguinte documentação do :

- Grupos de segurança do Amazon EC2 para instâncias do Linux no Guia do usuário do Amazon EC2
- Grupos de segurança do Amazon EC2 para instâncias do Windows no Guia do usuário do Amazon EC2
- Grupos de segurança do Amazon RD no Guia do usuário do Amazon RDS

WorkSpaces

No Amazon WorkSpaces, os usuários fazem login em um desktop com um nome de usuário e senha.

Para obter mais informações, consulte Conceitos básicos do WorkSpaces no Guia de administração do Amazon WorkSpaces.

Esses métodos de controle de acesso não fazem parte do IAM. O IAM permite controlar como esses produtos do AWS são administrados, criando ou terminando uma instância do Amazon

EC2, configurando novos desktops do WorkSpaces e assim por diante. Ou seja, o IAM ajuda a controlar as tarefas que são realizadas fazendo solicitações à Amazon Web Services e ajuda a controlar o acesso ao Console de gerenciamento da AWS. No entanto, o IAM não ajuda a gerenciar a segurança para tarefas, como fazer login em um sistema operacional (Amazon EC2), banco de dados (Amazon RDS) ou desktop (Amazon WorkSpaces).

Quando você trabalha com um produto específico da AWS, não deixe de ler a documentação para saber as opções de segurança para todos os recursos que pertencem a esse produto.

Usar o AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer fornece as seguintes capacidades:

- Os analisadores de acessos externos do IAM Access Analyzer ajudam a identificar os recursos da organização e as contas que são compartilhadas com uma entidade externa.
- Os analisadores de acesso interno do IAM Access Analyzer ajudam a identificar acesso interno a recursos selecionados da sua organização e das suas contas.
- Os analisadores de acessos não utilizados do IAM Access Analyzer ajudam a identificar os acessos não utilizados em sua organização e contas.
- O IAM Access Analyzer valida políticas do IAM em relação à gramática e a práticas recomendadas da AWS.
- As verificações de política personalizadas do IAM Access Analyzer ajudam a validar as políticas do IAM em relação aos padrões de segurança especificados.
- O IAM Access Analyzer gera políticas do IAM com base na atividade de acesso dos logs do AWS CloudTrail.

Identificar recursos compartilhados com uma entidade externa

O IAM Access Analyzer ajuda você a identificar os recursos em sua organização e suas contas, como buckets do Amazon S3 ou funções do IAM, que são compartilhados com uma entidade externa. Isso permite identificar o acesso não intencional aos seus recursos e dados, o que é um risco de segurança. O IAM Access Analyzer identifica recursos compartilhados com entidades externas usando raciocínio baseado em lógica para analisar as políticas baseadas em recurso no ambiente da AWS. Para cada instância de um recurso compartilhado fora de sua conta, o IAM Access Analyzer gera uma descoberta. As descobertas incluem informações sobre o acesso e a entidade externa a que é concedido. Elas podem ser analisadas para determinar se o acesso é intencional e seguro ou se não é intencional e representa um risco à segurança. Além de ajudar você a identificar recursos compartilhados com uma entidade externa, você pode usar as descobertas do IAM Access Analyzer para pré-visualizar como sua política afeta o acesso público e entre contas ao seu recurso antes de implantar as permissões do recurso. As descobertas são organizadas em um painel de resumo visual. O painel destaca a divisão entre as descobertas de acesso público e entre

contas e fornece um detalhamento das descobertas por tipo de recurso. Para saber mais sobre o painel, consulte Visualizar o painel de descobertas do IAM Access Analyzer.

> **Note**
>
> Uma entidade externa pode ser outra conta da AWS, um usuário-raiz, um usuário ou um perfil do IAM, um usuário federado, um usuário anônimo ou outra entidade que você possa usar para criar um filtro. Para obter mais informações, consulte Elementos da política JSON da AWS: principal.

Para obter instruções sobre como começar, consulte Criação de um analisador de acessos externos do IAM Access Analyzer. Depois de criar um analisador, você pode Gerenciamento de um analisador de acessos externos do IAM Access Analyzer.

Ao habilitar o IAM Access Analyzer, você cria um analisador para toda a sua organização ou sua conta. A organização ou a conta escolhida é conhecida como a zona de confiança do analisador. O analisador monitora todos os recursos compatíveis dentro da sua zona de confiança. Qualquer acesso aos recursos feito por entidades que estão dentro da sua zona de confiança é considerado confiável. Depois de habilitado, o IAM Access Analyzer analisará as políticas aplicadas a todos os recursos compatíveis da zona de confiança. Depois da primeira análise, o IAM Access Analyzer analisará essas políticas periodicamente. Se você adicionar uma nova política ou alterar uma política existente, o IAM Access Analyzer analisará a política nova ou atualizada em cerca de 30 minutos.

Ao analisar as políticas, se o IAM Access Analyzer identificar uma que conceda acesso a uma entidade principal externa que não esteja dentro da sua zona de confiança, ele gerará uma descoberta. Cada descoberta inclui detalhes sobre o recurso, sobre a entidade externa com acesso a ele e sobre as permissões concedidas para que você possa tomar as medidas apropriadas. É possível visualizar os detalhes incluídos na descoberta para determinar se o acesso ao recurso é intencional ou um risco potencial que deve ser resolvido. Quando você adiciona uma política a um recurso ou atualiza uma política existente, o IAM Access Analyzer analisa a política. O IAM Access Analyzer também analisa todas as políticas baseadas em recurso periodicamente.

Em raras ocasiões sob determinadas condições, o IAM Access Analyzer não recebe notificação sobre uma política adicionada ou atualizada, o que pode causar atrasos nas descobertas geradas. O IAM Access Analyzer poderá levar até seis horas para gerar ou resolver descobertas se você criar ou excluir um ponto de acesso multirregional associado a um bucket do Amazon S3 ou atualizar a política para o ponto de acesso multirregiões. Além disso, se houver um problema de entrega

com a entrega do log do AWS CloudTrail ou alterações de restrições da política de controle de recursos (RCP), a alteração da política não acionará uma nova verificação do recurso relatado na descoberta. Quando isso acontece, o IAM Access Analyzer analisa a política nova ou atualizada durante a próxima verificação periódica, que ocorre em até 24 horas. Se você desejar confirmar que uma alteração feita em uma política resolve um problema de acesso relatado em uma descoberta, verifique novamente o recurso relatado em uma descoberta usando o link Verificar novamente na página de detalhes Descobertas ou usando a operação `StartResourceScan` da API do IAM Access Analyzer. Para saber mais, consulte Resolver descobertas do IAM Access Analyzer.

> **Important**
>
> Para acesso externo, o IAM Access Analyzer analisa somente políticas aplicadas a recursos na mesma região da AWS em que está habilitado. Para monitorar todos os recursos do seu ambiente da AWS, é necessário criar um analisador de acesso externo para habilitar o IAM Access Analyzer em cada região em que você está usando recursos da AWS compatíveis. Para acessos não utilizados, as descobertas do analisador não mudam com base na região. Não é necessário criar um analisador de acesso sem uso em cada região em que você tem recursos.

O IAM Access Analyzer analisa acesso externo aos seguintes tipos de recursos:

- Buckets do Amazon Simple Storage Service
- Buckets de diretório do Amazon Simple Storage Service
- AWS Identity and Access ManagementFunções do para o
- AWS Key Management ServiceChaves do
- AWS LambdaFunções e camadas do
- Filas do Amazon Simple Queue Service
- AWS Secrets Managersegredos do
- Tópicos do Amazon Simple Notification Service
- Snapshots de volume do Amazon Elastic Block Store
- Snapshots de banco de dados do Amazon Relational Database Service
- Snapshots de cluster de banco de dados do Amazon Relational Database Service
- Repositórios do Amazon Elastic Container Registry

- Sistemas de arquivos do Amazon Elastic File System
- Amazon DynamoDB Streams
- Tabelas do Amazon DynamoDB

Identificar acesso interno a recursos críticos para os negócios

Para os recursos críticos para os negócios selecionados, o IAM Access Analyzer ajuda você a identificar quais entidades principais da sua organização ou conta têm acesso a eles. Essa análise é compatível com a implementação do princípio de privilégio mínimo, garantindo que os recursos especificados somente possam ser acessados pelas entidades principais pretendidas da sua organização.

Para obter instruções sobre como começar, consulte Criar um analisador de acesso interno do IAM Access Analyzer. Depois de criar um analisador, você pode Gerenciar um analisador de acesso interno do IAM Access Analyzer.

A análise de acesso interno ajuda você a:

- Determinar quais usuários ou perfis do IAM da sua conta ou organização podem acessar os recursos especificados
- Entender os caminhos de acesso entre entidades principais e recursos no ambiente da AWS
- Verificar se os controles de acesso estão funcionando como pretendido
- Revisar descobertas para determinar se o acesso é intencional e seguro ou se o acesso não é intencional e representa um risco à segurança
- Identificar e corrigir acesso não intencional na organização

O IAM Access Analyzer analisa acesso interno aos seguintes tipos de recursos:

- Buckets do Amazon Simple Storage Service
- Buckets de diretório do Amazon Simple Storage Service
- Snapshots de banco de dados do Amazon Relational Database Service
- Snapshots de cluster de banco de dados do Amazon Relational Database Service
- Amazon DynamoDB Streams
- Tabelas do Amazon DynamoDB

Identificação de acessos não utilizados concedidos a perfis e usuários do IAM

O IAM Access Analyzer ajuda você a identificar e analisar acessos não utilizados em sua organização e contas da AWS. O IAM Access Analyzer monitora continuamente todas os usuários e perfis do IAM em sua organização e contas AWS e gera descobertas para acessos não utilizados. As descobertas destacam perfis não utilizados, chaves de acesso não utilizadas para usuários do IAM e senhas não utilizadas para usuários do IAM. Para funções e usuários ativos do IAM, as descobertas fornecem visibilidade sobre serviços e ações não utilizados.

Para obter instruções sobre como criar um analisador de acesso não usado e gerenciar suas descobertas, consulte Gerenciamento de um analisador de acessos não utilizados do IAM Access Analyzer.

O IAM Access Analyzer analisa as informações do último acesso de todas as funções em sua organização e contas AWS para ajudar você a identificar acessos não utilizados. As informações do último acesso da ação do IAM ajudam a identificar ações não utilizadas para perfis em suas Contas da AWS. Para obter mais informações, consulte Refinar permissões na AWS usando informações do último acesso.

As descobertas dos analisadores de acesso externo, interno e não utilizado são organizadas em um painel de resumo visual. O painel destaca os recursos da AWS e as Contas da AWS que têm mais descobertas, e fornece um detalhamento das descobertas por tipo. Para obter mais informações sobre o painel, consulte Visualizar o painel de descobertas do IAM Access Analyzer.

Validação de políticas em relação às práticas recomendadas da AWS

É possível validar suas políticas em relação à gramática da política do IAM e às práticas recomendadas da AWS usando as verificações básicas de políticas fornecidas pela validação de políticas do IAM Access Analyzer. É possível criar ou editar uma política usando a AWS CLI, a API da AWS ou o editor de políticas de JSON no console do IAM. É possível visualizar as descobertas de verificação de validação de política que incluem avisos de segurança, erros, avisos gerais e sugestões para sua política. Essas descobertas fornecem recomendações práticas que ajudam a criar políticas que sejam funcionais e estejam em conformidade com as práticas recomendadas da AWS. Para obter mais informações sobre como validar políticas usando validação de política,

consulte Validar políticas com o IAM Access Analyzer. Para obter uma lista das verificações de políticas que o IAM Access Analyzer executa, consulte Verificações de validação de políticas.

Validar políticas de acordo com seus padrões especificados de segurança

É possível validar suas políticas de acordo com seus padrões especificados de segurança usando verificações de política personalizadas do IAM Access Analyzer. É possível criar ou editar uma política usando a AWS CLI, a API da AWS ou o editor de políticas de JSON no console do IAM. Por meio do console, você pode verificar se sua política atualizada concede novo acesso em comparação com a versão existente. Por meio da AWS CLI e da API AWS, você também pode verificar se ações específicas do IAM que você considera críticas não são permitidas por uma política. Essas verificações destacam uma instrução de política que concede novo acesso. É possível atualizar a declaração de política e executar novamente as verificações até que a política esteja em conformidade com seu padrão de segurança. Para obter mais informações sobre como validar políticas usando verificações de política personalizadas, consulte Validar políticas com verificações de políticas do IAM Access Analyzer.

Geração de políticas

O IAM Access Analyzer analisa seus logs do AWS CloudTrail para identificar ações e serviços que foram usados por uma entidade do IAM (usuário ou função) dentro do intervalo de datas especificado. Em seguida, ele gera uma política do IAM com base nessa atividade de acesso. É possível usar a política gerada para refinar as permissões de uma entidade anexando-a a um usuário ou um perfil do IAM. Para saber mais sobre como gerar políticas usando o IAM Access Analyzer, consulte Geração de política do IAM Access Analyzer.

Preços do IAM Access Analyzer

O IAM Access Analyzer cobra pela análise de acessos não utilizados com base no número de funções e usuários do IAM analisados por analisador por mês.

- Você será cobrado por cada analisador de acessos não utilizados que criar.
- A criação de analisadores de acessos não utilizados em várias regiões resultará na cobrança por cada analisador.

- Os perfis vinculados ao serviço não são analisados quanto à atividade de acesso não utilizada e não são incluídos no número total de perfis do IAM analisados.

O IAM Access Analyzer cobra a análise de acesso interno com base no número de recursos monitorados por analisador de acesso interno por mês.

O IAM Access Analyzer cobra por verificações de política personalizadas com base no número de solicitações de API feitas ao IAM Access Analyzer para verificar novos acessos.

Para obter uma lista completa de preços e cobranças do IAM Access Analyzer, consulte Preços do IAM Access Analyzer.

Para ver sua fatura, acesse o Painel de gerenciamento de custos e faturamento no console do Gerenciamento de Faturamento e Custos da AWS. Sua fatura contém links para relatórios de uso que fornecem detalhes sobre sua conta. Para saber mais sobre o faturamento da Conta da AWS, consulte o Guia do usuário do AWS Billing.

Se tiver dúvidas sobre faturamento, contas e eventos da AWS, entre em contato com o Suporte.

Descobertas do IAM Access Analyzer

O IAM Access Analyzer gera descobertas para acesso externo, acesso interno e acesso não utilizado na sua Conta da AWS ou organização.

Para acessos externos, o IAM Access Analyzer gera uma descoberta para cada instância de uma política baseada em recursos que concede acesso a um recurso dentro da sua zona de confiança para uma entidade principal que não esteja dentro da sua zona de confiança. Ao criar um analisador de acessos externos, você escolhe uma organização ou uma Conta da AWS para analisar. Qualquer principal na organização ou na conta que você escolher para o analisador é considerado confiável. Como as entidades principais na mesma organização ou conta são confiáveis, os recursos e as entidades principais dentro da organização ou da conta compreendem a zona de confiança do analisador. Qualquer compartilhamento dentro da zona de confiança é considerado seguro, portanto, o IAM Access Analyzer não gera uma descoberta. Por exemplo, se você escolher uma organização como a zona de confiança de um analisador, todos os recursos e as entidades principais da organização estarão dentro da zona de confiança. Se você conceder permissões a um bucket do Amazon S3 em uma das contas-membro da organização para uma entidade principal em outra conta-membro da organização, o IAM Access Analyzer não gerará uma descoberta. No entanto, se você conceder permissão a uma entidade principal em uma conta que não seja membro da organização, o IAM Access Analyzer gerará uma descoberta.

Para acesso interno, o IAM Access Analyzer gera descobertas quando há um possível caminho de acesso entre um perfil ou usuário do IAM da sua organização e os recursos especificados. Como acontece com a análise de acesso externo, o escopo escolhido (organização ou conta) determina o que é considerado interno. Se você selecionar uma organização como escopo, o IAM Access Analyzer gerará descobertas para caminhos de acesso entre as entidades principais e os recursos da sua organização. Se você selecionar uma conta, as descobertas serão geradas para os caminhos de acesso dentro dessa conta específica. O IAM Access Analyzer usa raciocínio automatizado para avaliar todas as políticas do IAM e monitorar quem tem acesso aos recursos.

A combinação das descobertas de acesso externo e interno com a mesma zona de confiança fornece uma análise abrangente de todo acesso possível a um determinado recurso, tanto de dentro quanto de fora dos limites de confiança definidos.

Para acesso não utilizado, o IAM Access Analyzer gera descobertas para o acesso não utilizado concedido na sua organização e nas suas contas da AWS. Quando você cria um analisador de acessos não utilizados, o IAM Access Analyzer monitora continuamente todas as funções e usuários do IAM em sua organização e contas AWS e gera descobertas sobre acessos não utilizados. O IAM Access Analyzer gera os seguintes tipos de descobertas para acessos não utilizados:

- Perfis não utilizados: perfis sem atividade de acesso dentro da janela de uso especificada.
- Chaves de acesso e senhas de usuários do IAM não usadas: credenciais pertencentes a usuários do IAM que não foram usadas para acessar sua Conta da AWS durante a janela de uso especificada.
- Permissões não utilizadas: permissões de nível de serviço e de ação que não foram usadas por um perfil na janela de uso especificada. O IAM Access Analyzer usa políticas baseadas em identidade anexadas às funções para determinar os serviços e ações que essas funções podem acessar. O IAM Access Analyzer oferece suporte à análise de permissões não utilizadas para todas as permissões de nível de serviço. Para obter uma lista completa das permissões de nível de ação suportadas para descobertas de acessos não utilizados, consulte Serviços e ações para os quais a ação do IAM acessou informações pela última vez.

> **Note**
>
> O IAM Access Analyzer oferece descobertas de acesso externo gratuitamente. As descobertas de acesso não utilizado são cobradas com base no número de perfis e usuários do IAM analisados por analisador por mês. Também são cobradas as descobertas de acesso

interno com base no número de recursos da AWS monitorados por analisador por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Tópicos

- Como as descobertas do IAM Access Analyzer funcionam
- Conceitos básicos do AWS Identity and Access Management Access Analyzer
- Visualizar o painel de descobertas do IAM Access Analyzer
- Revisar descobertas do IAM Access Analyzer
- Filtrar descobertas do IAM Access Analyzer
- Arquivar descobertas do IAM Access Analyzer
- Resolver descobertas do IAM Access Analyzer
- Descobertas de erros do IAM Access Analyzer
- Tipos de recursos compatíveis com o IAM Access Analyzer para acesso externo e interno
- Administrador delegado do IAM Access Analyzer
- Regras de arquivamento
- Monitoramento do AWS Identity and Access Management Access Analyzer com o Amazon EventBridge
- Integrar o IAM Access Analyzer com o AWS Security Hub CSPM
- Registrar em log chamadas de API do IAM Access Analyzer com o AWS CloudTrail
- Chaves de filtro do IAM Access Analyzer
- Usar perfis vinculados a serviço do AWS Identity and Access Management Access Analyzer

Como as descobertas do IAM Access Analyzer funcionam

Este tópico descreve os conceitos e os termos usados no IAM Access Analyzer para ajudar você a se familiarizar com a maneira como o IAM Access Analyzer monitora o acesso aos seus recursos da AWS.

Descobertas de acessos externos

As descobertas de acesso externo são geradas somente uma vez para cada instância de um recurso compartilhado fora da sua zona de confiança. Sempre que uma política baseada em recursos é

modificada, o IAM Access Analyzer analisa a política. Se a política atualizada compartilhar um recurso já identificado em uma descoberta, mas com permissões ou condições diferentes, será gerada outra descoberta para essa instância do compartilhamento de recursos. Alterações em uma política de controle de recursos que afetam a restrição da política de controle de recursos (RCP) também geram uma nova descoberta. O IAM Access Analyzer também avalia as configurações de controle de acesso estabelecidas por políticas declarativas. Se o acesso na primeira descoberta for removido, essa descoberta será atualizada para um status de Resolvida.

O status de todas as descobertas permanece como Ativa até que sejam arquivadas ou até que o acesso que gerou a descoberta seja removido. Ao remover o acesso, o status da descoberta é atualizado para Resolvida.

> **Note**
>
> Pode levar até 30 minutos depois que uma política é modificada para que o IAM Access Analyzer analise o recurso e atualize a descoberta de acesso externo. Alterações em uma política de controle de recursos (RCP) não acionam uma nova verificação do recurso relatado na descoberta. O IAM Access Analyzer analisa a política nova ou atualizada durante a próxima verificação periódica, que ocorre em até 24 horas.

Como o IAM Access Analyzer gera descobertas para acessos externos

O AWS Identity and Access Management Access Analyzer usa uma tecnologia chamada Zelkova para analisar as políticas do IAM e identificar o acesso externo aos recursos.

O Zelkova converte políticas do IAM em declarações lógicas equivalentes e as executa por meio de um conjunto de solucionadores lógicos especializados e de uso geral (teorias do módulo da satisfatibilidade). O IAM Access Analyzer aplica o Zelkova repetidamente a uma política, usando consultas cada vez mais específicas para caracterizar os tipos de acesso que a política permite com base em seu conteúdo. Para obter mais informações sobre as teorias do módulo da satisfatibilidade, consulte Satisfiability Modulo Theories.

Para analisadores de acessos externos, o IAM Access Analyzer não examina os logs de acesso para determinar se uma entidade externa realmente acessou um recurso dentro da sua zona de confiança. Em vez disso, ele gera uma descoberta quando uma política baseada em recursos permite o acesso a um recurso, independentemente de o recurso ter sido acessado pela entidade externa.

Além disso, o IAM Access Analyzer não considera o estado de nenhuma conta externa ao fazer suas determinações. Se ele indicar que a conta 111122223333 pode acessar o seu bucket do Amazon S3, ele não terá nenhuma informação sobre os usuários, funções, políticas de controle de serviços (SCP) ou outras configurações relevantes nessa conta. Isso é para a privacidade do cliente, pois o IAM Access Analyzer não sabe quem é o proprietário da outra conta. Isso também é para fins de segurança, pois é importante conhecer o possível acesso externo, mesmo que atualmente não haja entidades principais ativas que possam usá-lo.

O IAM Access Analyzer considera apenas determinadas chaves de condição do IAM que os usuários externos não podem influenciar diretamente ou que, de outra forma, têm impacto na autorização. Para obter exemplos de chaves de condição que o IAM Access Analyzer considera, consulte Chaves de filtro do IAM Access Analyzer.

Atualmente, o IAM Access Analyzer não relata descobertas de entidades principais de AWS service (Serviço da AWS) ou contas de serviço interno. Nos raros casos em que não é possível determinar totalmente se uma declaração de política concede acesso a uma entidade externa, ele erra ao declarar uma descoberta falso positiva. Isso ocorre porque o IAM Access Analyzer foi projetado para fornecer uma visão abrangente do compartilhamento de recursos em sua conta e para minimizar os falsos negativos.

Descobertas de acesso interno

Para usar análise de acesso interno, você deve primeiro configurar o analisador selecionando os recursos específicos que deseja monitorar. Depois de configuradas, as descobertas de acesso interno são geradas quando uma entidade principal (usuário ou perfil do IAM) da sua organização ou conta tem acesso aos recursos selecionados. Uma nova descoberta será gerada a próxima vez que o analisador examinar os recursos especificados e identificar uma entidade principal que tenha acesso aos recursos. Se uma política atualizada permitir uma entidade principal já identificada em uma descoberta, mas com permissões ou condições diferentes, uma nova descoberta será gerada para essa instância de entidade principal e recurso. Essa política atualizada pode ser uma política baseada em recurso, uma política baseada em identidade, uma política de controle de serviços (SCP) ou uma política de controle de recursos (RCP). O IAM Access Analyzer também avalia as configurações de controle de acesso estabelecidas por políticas declarativas.

> **Note**
>
> As descobertas de acesso interno somente estão disponíveis usando a ação da API ListFindingsV2.

Como o IAM Access Analyzer gera descobertas para acesso interno

Para analisar acesso interno, você deve criar um analisador separado para descobertas de acesso interno para seus recursos, mesmo que já tenha criado um analisador para gerar descobertas de acesso externo ou descobertas de acesso não utilizado.

Depois de criar o analisador de acesso interno, o IAM Access Analyzer avalia todas as políticas baseadas em recurso, políticas baseadas em identidade, políticas de controle de serviços (SCPs), políticas de controle de recursos (RCPs) e limites de permissões da conta ou organização especificada.

Criando um analisador dedicado a acesso interno aos recursos selecionados, você pode identificar:

- Quando uma entidade principal da sua organização ou conta pode acessar os recursos selecionados
- O total de permissões permitidas de fato para uma entidade principal com base na interseção de todas as políticas aplicáveis
- Caminhos de acesso complexos em que uma entidade principal obtém acesso com base na combinação de políticas de identidades e políticas de recursos

> **Note**
>
> O IAM Access Analyzer não pode gerar descobertas de acesso interno para organizações com mais de 70.000 entidades principais (usuários e perfis do IAM combinados).

Descobertas de acessos não utilizados

As descobertas de acesso não utilizado são geradas para entidades (entidades principais) do IAM na conta ou organização selecionada com base no número de dias especificado durante a criação do analisador. Uma nova descoberta será gerada na próxima vez que o analisador examinar as entidades se uma das seguintes condições for atendida:

- Uma função fica inativa durante o número especificado de dias.
- Uma permissão não utilizada, uma senha de usuário não utilizada ou uma chave de acesso de usuário não utilizada ultrapassam o número especificado de dias.

> **Note**
>
> As descobertas de acesso não utilizadas só estão disponíveis usando a ação da API ListFindingsV2.

Como o IAM Access Analyzer gera descobertas para acessos não utilizados

Para analisar acesso não utilizado, você deve criar um analisador separado para descobertas de acesso não utilizado para os perfis, mesmo que já tenha criado um analisador para gerar descobertas de acesso externo ou interno para os recursos.

Depois de criar o analisador de acessos não utilizados, o IAM Access Analyzer revisa a atividade de acesso para identificar acessos não utilizados. O IAM Access Analyzer examina as últimas informações acessadas para todos os usuários do IAM, perfis do IAM – incluindo perfis de serviço, chaves de acesso e senhas de usuário em toda a organização e em todas as suas contas da AWS. Isso ajuda você a identificar o acesso não utilizado.

> **Note**
>
> Um perfil vinculado ao serviço é um tipo especial de perfil de serviço vinculado a um AWS service (Serviço da AWS) e pertencente ao serviço. Os perfis vinculados ao serviço não são analisados por analisadores de acesso não utilizados.

Para perfis e usuários ativos do IAM, o IAM Access Analyzer usa as últimas informações acessadas para serviços e ações do IAM para identificar permissões não utilizadas. Isso permite que você escale seu processo de revisão para todas as organizações e contas da AWS. Também é possível usar as informações do último acesso da ação para realizar uma investigação mais profunda dos perfis individuais. Isso fornece informações mais granulares sobre quais permissões específicas não estão sendo utilizadas.

Ao criar um analisador dedicado ao acesso não utilizado, você pode analisar e identificar de forma abrangente o acesso não utilizado em seu ambiente da AWS, complementando as descobertas geradas pelo seu analisador de acesso externo existente.

Conceitos básicos do AWS Identity and Access Management Access Analyzer

Use as informações deste tópico para saber mais sobre os requisitos necessários para usar e gerenciar o AWS Identity and Access Management Access Analyzer.

Se estiver usando o IAM Access Analyzer em uma organização do AWS Organizations, você pode Adicionar um administrador delegado ao IAM Access Analyzer. Para obter informações sobre os limites de serviço, consulte Cotas do IAM Access Analyzer.

Permissões necessárias para usar o IAM Access Analyzer

Para configurar e usar o IAM Access Analyzer com êxito, a conta usada deve receber as permissões necessárias.

AWSPolíticas gerenciadas pela para o IAM Access Analyzer

O AWS Identity and Access Management Access Analyzer fornece políticas gerenciadas da AWS para ajudar você a começar rapidamente.

- IAMAccessAnalyzerFullAccess: permite acesso total dos administradores ao IAM Access Analyzer. Esta política também permite criar as funções vinculadas ao serviço que são necessárias para permitir que o IAM Access Analyzer analise recursos em sua conta ou organização da AWS.
- IAMAccessAnalyzerReadOnlyAccess: permite acesso somente leitura ao IAM. Você deve adicionar políticas adicionais às suas identidades do IAM (usuários, grupos de usuários ou funções) para permitir que elas visualizem suas descobertas.

Recursos definidos pelo IAM Access Analyzer

Para visualizar os recursos definidos pelo Access Analyzer, consulte Tipos de recursos definidos pelo IAM Access Analyzer na Referência de autorização do serviço.

Permissões necessárias do serviço IAM Access Analyzer

O IAM Access Analyzer usa uma função vinculada ao serviço (SRL) chamada de `AWSServiceRoleForAccessAnalyzer`. Essa SLR concede ao serviço acesso somente leitura a fim de analisar recursos AWS com políticas baseadas em recursos e analisar acessos não utilizados em seu nome. O serviço cria a função na sua conta nas seguintes situações:

- Você cria um analisador de acessos externos com sua conta como zona de confiança.
- Você cria um analisador de acessos não utilizados com sua conta como a conta selecionada.
- Você cria um analisador de acesso interno com a sua conta como zona de confiança.

Para obter mais informações, consulte Usar perfis vinculados a serviço do AWS Identity and Access Management Access Analyzer.

> **Note**
>
> O IAM Access Analyzer é regional. Para analisadores de acesso externo e interno, você deve habilitar o IAM Access Analyzer em cada região separadamente.
> Para acessos não utilizados, as descobertas do analisador não mudam com base na região. Não é necessário criar um analisador em cada região em que você tem recursos.

Em alguns casos, após você criar um analisador no IAM Access Analyzer, a página ou o painel Descobertas é carregado sem nenhuma descoberta ou resumo. Isso pode ocorrer devido a um atraso no console para preencher as descobertas. Talvez seja necessário atualizar manualmente o navegador ou voltar mais tarde para visualizar as descobertas ou o resumo. Se ainda não for exibida nenhuma descoberta para um analisador de acessos externos, é porque você não tem recursos compatíveis na conta que possam ser acessados por uma entidade externa. Se uma política que concede acesso a uma entidade externa for aplicada a um recurso, o IAM Access Analyzer gerará uma descoberta.

> **Note**
>
> Para analisadores de acesso externo, depois que uma política é modificada, o IAM Access Analyzer pode levar até 30 minutos para analisar o recurso e gerar uma nova descoberta ou atualizar uma descoberta existente para o acesso ao recurso.
> Quando você cria um analisador de acesso interno, pode levar vários minutos ou horas para as descobertas ficarem disponíveis. Após a verificação inicial, o IAM Access Analyzer torna a varrer automaticamente todas as políticas a cada 24 horas.
> Para todos os tipos de analisadores de acesso, as atualizações das descobertas podem não ser refletidas imediatamente no painel.

Permissões necessárias do IAM Access Analyzer para visualizar o painel de descobertas

Para visualizar o painel de descobertas do IAM Access Analyzer, a conta usada deve receber acesso a fim de realizar as seguintes ações necessárias:

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Para visualizar todas as ações definidas pelo IAM Access Analyzer, consulte Ações definidas pelo IAM Access Analyzer na Referência de autorização do serviço.

Status do IAM Access Analyzer

Para visualizar o status dos analisadores, selecione Analyzers (Analisadores). Os analisadores criados para uma organização ou uma conta podem ter os seguintes status:

Status	Descrição
Ativo	Para analisadores de acesso externo e interno, o analisador está monitorando ativamente os recursos dentro da sua zona de confiança. O analisador gera ativamente novas descobertas e atualiza as descobertas existentes. Para analisadores de acessos não utilizados, o analisador está monitorando ativamente os acessos não utilizados na Conta da AWS ou na organização selecionada no período de rastreamento especificado. O analisador gera ativamente novas descobertas e atualiza as descobertas existentes.
Criando	A criação do analisador ainda está em andamento. O analisador fica ativo quando a criação é concluída.

Status	Descrição
Desabilitado	O analisador é desabilitado devido a uma ação executada pelo administrador do AWS Organizations. Por exemplo, remover a conta do analisador como administrador delegado do IAM Access Analyzer. Quando o analisador está em um estado desabilitado, ele não gera novas descobertas nem atualiza as descobertas existentes.
Failed	A criação do analisador falhou devido a um problema de configuração. O analisador não gerará nenhuma descoberta. Exclua o analisador e crie outro analisador.

Criação de um analisador de acessos externos do IAM Access Analyzer

Para habilitar um analisador de acessos externos em uma região, é necessário criar um analisador nessa região. É necessário criar um analisador de acessos externos em cada região onde deseja monitorar o acesso aos recursos.

> **Note**
>
> Após você criar ou atualizar um analisador, pode levar algum tempo para as descobertas ficarem disponíveis.

Criação de um analisador de acessos externos com a Conta da AWS como zona de confiança

1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Selecione Create analyzer (Criar analisador).
4. Na seção Análise, escolha Análise de recursos: acesso externo.
5. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.

6. Insira um nome para o analisador.
7. Escolha Conta atual como a zona de confiança do analisador.

> **Note**
>
> Se sua conta não for a conta de gerenciamento do AWS Organizations ou uma conta de administrador substituto, você pode criar apenas um analisador com sua conta como a zona de confiança.

8. Opcional. Adicione as tags que deseja aplicar ao analisador.
9. Selecione Create analyzer (Criar analisador).

Ao criar um analisador de acessos externos para habilitar o IAM Access Analyzer, uma função vinculada a serviço chamada `AWSServiceRoleForAccessAnalyzer` será criada em sua conta.

Criação de um analisador de acessos externos com a organização como zona de confiança

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Selecione Create analyzer (Criar analisador).
4. Na seção Análise, escolha Análise de recursos: acesso externo.
5. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
6. Insira um nome para o analisador.
7. Escolha Organização atual como a zona de confiança para o analisador.
8. Opcional. Adicione as tags que deseja aplicar ao analisador.
9. Selecione Enviar.

Quando você cria um analisador de acessos externos com a organização como a zona de confiança, uma função vinculada a serviço chamada `AWSServiceRoleForAccessAnalyzer` é criada em cada conta da organização.

Gerenciamento de um analisador de acessos externos do IAM Access Analyzer

Para habilitar um analisador de acessos externos em uma região, é necessário criar um analisador nessa região. É necessário criar um analisador de acessos externos em cada região onde deseja monitorar o acesso aos recursos.

> **Note**
>
> Após você criar ou atualizar um analisador, pode levar algum tempo para as descobertas ficarem disponíveis.

Atualização de um analisador de acessos externos

Use o procedimento a seguir para atualizar um analisador de acesso externo.

1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Na seção Analisadores, escolha o nome do analisador de acesso externo a ser gerenciado.
4. Na guia Regras de arquivamento, é possível criar, editar ou excluir regras de arquivamento para o analisador. Para obter mais informações, consulte Regras de arquivamento.
5. Na guia Tags, é possível gerenciar e criar tags para o analisador. Para obter mais informações, consulte Tags para recursos do AWS Identity and Access Management.

Exclusão de um analisador de acessos externos

Use o procedimento a seguir para excluir um analisador de acesso externo. Quando você exclui um analisador, os recursos não são mais monitorados e nenhuma nova descoberta é gerada. Todas as descobertas geradas pelo analisador são excluídas.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Na seção Analisadores, escolha o nome do analisador de acesso externo a ser excluído.
4. Escolha Excluir analisador.
5. Insira excluir e escolha Excluir para confirmar a exclusão do analisador.

Criar um analisador de acesso interno do IAM Access Analyzer

Para habilitar um analisador de acesso interno em uma região, você deve criar um analisador nessa região. Você deve criar um analisador de acesso interno em cada região onde quiser monitorar o acesso aos recursos.

O IAM Access Analyzer cobra análise de acesso interno com base no número de recursos monitorados por analisador por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

> **Note**
>
> Após você criar ou atualizar um analisador, pode levar algum tempo para as descobertas ficarem disponíveis.
> O IAM Access Analyzer não pode gerar descobertas de acesso interno para organizações com mais de 70.000 entidades principais (usuários e perfis do IAM combinados).
> Você pode criar somente um analisador de acesso interno ao nível organizacional de uma organização da AWS.

Criar um analisador de acesso interno com a Conta da AWS como zona de confiança

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Selecione Create analyzer (Criar analisador).
4. Na seção Análise, escolha Análise de recursos: acesso interno.
5. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
6. Insira um nome para o analisador.
7. Escolha Conta atual como a zona de confiança do analisador.

> **Note**
>
> Se sua conta não for a conta de gerenciamento do AWS Organizations ou uma conta de administrador substituto, você pode criar apenas um analisador com sua conta como a zona de confiança.

8. Na seção Recursos a serem analisados, adicione os recursos para o analisador monitorar.

   - Para adicionar recursos por conta, escolha Adicionar > Adicionar recursos das contas selecionadas.

     1. Escolha Todos os tipos de recursos compatíveis ou Definir tipos de recursos específicos e selecione os tipos de recursos na lista Tipo de recurso.

        Os analisadores de acesso interno são compatíveis com os seguintes tipos de recursos:

        - Buckets do Amazon Simple Storage Service
        - Buckets de diretório do Amazon Simple Storage Service
        - Snapshots de banco de dados do Amazon Relational Database Service
        - Snapshots de cluster de banco de dados do Amazon Relational Database Service
        - Amazon DynamoDB Streams
        - Tabelas do Amazon DynamoDB

     2. Selecione Adicionar recursos.

   - Para adicionar recursos pelo nome do recurso da Amazon (ARN), escolha Adicionar recursos > Adicionar recursos colando o ARN do recurso.

     > **Note**
     >
     > ARNs devem ser correspondências exatas – curingas não são compatíveis. Para o Amazon S3, somente ARNs de bucket são compatíveis. ARNs de objetos e prefixos do Amazon S3 não são compatíveis.

     1. Para cada ARN de recurso, insira o ID do proprietário da conta e o ARN do recurso, separados por vírgula. Insira um único ID de proprietário de conta e ARN de recurso por linha.
     2. Selecione Adicionar recursos.

   - Para adicionar recursos por meio de um arquivo CSV, escolha Adicionar recursos > Adicionar recursos carregando um CSV.

     Você pode usar o Explorador de recursos da AWS para pesquisar recursos em suas contas e exportar um arquivo CSV. Depois, você pode carregar o arquivo CSV para configurar os recursos para o analisador monitorar.

    1. Escolha Escolher arquivo e selecione o arquivo CSV no seu computador.
    2. Selecione Adicionar recursos.
9. Opcional. Adicione as tags que deseja aplicar ao analisador.
10. Selecione Create analyzer (Criar analisador).

Quando você cria um analisador de acesso interno para habilitar o IAM Access Analyzer, um perfil vinculado ao serviço denominado `AWSServiceRoleForAccessAnalyzer` é criado na sua conta.

Criar um analisador de acesso interno com a organização como zona de confiança

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Selecione Create analyzer (Criar analisador).
4. Na seção Análise, escolha Análise de recursos: acesso interno.
5. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
6. Insira um nome para o analisador.
7. Escolha Toda a organização como a zona de confiança do analisador.
8. Na seção Recursos a serem analisados, adicione os recursos para o analisador monitorar.

    - Para adicionar recursos à conta, escolha Adicionar recursos > Adicionar recursos das contas selecionadas.
        1. Escolha Todos os tipos de recursos compatíveis ou Definir tipos de recursos específicos e selecione os tipos de recursos na lista Tipo de recurso.

            Os analisadores de acesso interno são compatíveis com os seguintes tipos de recursos:

            - Buckets do Amazon Simple Storage Service
            - Buckets de diretório do Amazon Simple Storage Service
            - Snapshots de banco de dados do Amazon Relational Database Service
            - Snapshots de cluster de banco de dados do Amazon Relational Database Service
            - Amazon DynamoDB Streams
            - Tabelas do Amazon DynamoDB

2. Para selecionar contas da sua organização, escolha Selecionar na organização. Na seção Selecionar contas, escolha Hierarquia para selecionar contas por estrutura organizacional ou Lista para selecionar contas em uma lista de todas as contas da organização.

   Para inserir manualmente contas da organização, escolha Inserir ID da conta da AWS. Insira um ou mais IDs de Conta da AWS separados por vírgulas no campo ID da conta da AWS.

3. Selecione Adicionar recursos.

- Para adicionar recursos pelo nome do recurso da Amazon (ARN), escolha Adicionar recursos > Adicionar recursos colando o ARN do recurso.

  > **Note**
  >
  > ARNs devem ser correspondências exatas – curingas não são compatíveis. Para o Amazon S3, somente ARNs de bucket são compatíveis. ARNs de objetos e prefixos do Amazon S3 não são compatíveis.

  1. Para cada ARN de recurso, insira o ID do proprietário da conta e o ARN do recurso, separados por vírgula. Insira um único ID de proprietário de conta e ARN de recurso por linha.
  2. Selecione Adicionar recursos.

- Para adicionar recursos por meio de um arquivo CSV, escolha Adicionar recursos > Adicionar recursos carregando um CSV.

  Você pode usar o Explorador de recursos da AWS para pesquisar recursos em suas contas e exportar um arquivo CSV. Depois, você pode carregar o arquivo CSV para configurar os recursos para o analisador monitorar.

  1. Escolha Escolher arquivo e selecione o arquivo CSV no seu computador.
  2. Selecione Adicionar recursos.

9. Opcional. Adicione as tags que deseja aplicar ao analisador.
10. Selecione Enviar.

Quando você cria um analisador de acesso interno com a organização como a zona de confiança, um perfil vinculado ao serviço, denominado `AWSServiceRoleForAccessAnalyzer`, é criado em cada conta da organização.

Gerenciar um analisador de acesso interno do IAM Access Analyzer

Para habilitar um analisador de acesso interno em uma região, você deve criar um analisador nessa região. Você deve criar um analisador de acesso interno em cada região onde quiser monitorar o acesso aos recursos.

> **Note**
>
> Após você criar ou atualizar um analisador, pode levar algum tempo para as descobertas ficarem disponíveis.

Atualizar um analisador de acesso interno

Use o procedimento a seguir para atualizar um analisador de acesso interno.

1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Na seção Analisadores, escolha o nome do analisador de acesso interno a ser gerenciado.
4. Na guia Regras de arquivamento, é possível criar, editar ou excluir regras de arquivamento para o analisador. Para obter mais informações, consulte Regras de arquivamento.
5. Na guia Tags, é possível gerenciar e criar tags para o analisador. Para obter mais informações, consulte Tags para recursos do AWS Identity and Access Management.
6. Na guia Recursos, escolha Editar na seção Recursos a serem analisados.

    a. Para adicionar recursos por conta, escolha Adicionar recursos > Adicionar recursos das contas selecionadas.

        1. Escolha Todos os tipos de recursos compatíveis ou Definir tipos de recursos específicos e selecione os tipos de recursos na lista Tipo de recurso.

            Os analisadores de acesso interno são compatíveis com os seguintes tipos de recursos:

            - Buckets do Amazon Simple Storage Service
            - Buckets de diretório do Amazon Simple Storage Service

- Snapshots de banco de dados do Amazon Relational Database Service
- Snapshots de cluster de banco de dados do Amazon Relational Database Service
- Amazon DynamoDB Streams
- Tabelas do Amazon DynamoDB

2. Selecione Adicionar recursos.

b. Para adicionar recursos pelo nome do recurso da Amazon (ARN), escolha Adicionar recursos > Adicionar recursos colando o ARN do recurso.

> **Note**
>
> ARNs devem ser correspondências exatas – curingas não são compatíveis. Para o Amazon S3, somente ARNs de bucket são compatíveis. ARNs de objetos e prefixos do Amazon S3 não são compatíveis.

1. Para cada ARN de recurso, insira o ID do proprietário da conta e o ARN do recurso, separados por vírgula. Insira um único ID de proprietário de conta e ARN de recurso por linha.
2. Selecione Adicionar recursos.

c. Para adicionar recursos por meio de um arquivo CSV, escolha Adicionar recursos > Adicionar recursos carregando um CSV.

Você pode usar o Explorador de recursos da AWS para pesquisar recursos em suas contas e exportar um arquivo CSV. Depois, você pode carregar o arquivo CSV para configurar os recursos para o analisador monitorar.

1. Escolha Escolher arquivo e selecione o arquivo CSV no seu computador.
2. Selecione Adicionar recursos.

d. Para remover recursos do analisador, marque a caixa de seleção ao lado dos recursos a serem removidos e escolha Remover.

e. Escolha Salvar alterações.

> **Note**
>
> Qualquer atualização do analisador será avaliada na próxima varredura automática dentro de 24 horas.

Excluir um analisador de acesso interno

Use o procedimento a seguir para excluir um analisador de acesso interno. Quando você exclui um analisador, os recursos não são mais monitorados e nenhuma nova descoberta é gerada. Todas as descobertas geradas pelo analisador são excluídas.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Na seção Analisadores, escolha o nome do analisador de acesso interno a ser excluído.
4. Escolha Excluir analisador.
5. Insira excluir e escolha Excluir para confirmar a exclusão do analisador.

Criação de um analisador de acessos não utilizados do IAM Access Analyzer

Criação de um analisador de acessos não utilizados para a conta atual

Use o procedimento a seguir para criar um analisador de acessos não utilizados para uma única Conta da AWS. Para acessos não utilizados, as descobertas do analisador não mudam com base na região. Não é necessário criar um analisador em cada região em que você tem recursos.

O IAM Access Analyzer cobra pela análise de acessos não utilizados com base no número de usuários e perfis do IAM analisados por mês por analisador. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

> **Note**
>
> Após você criar ou atualizar um analisador, pode levar algum tempo para as descobertas ficarem disponíveis.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.

2. Em Analisador de acessos, escolha Configurações do analisador.
3. Selecione Create analyzer (Criar analisador).
4. Na seção Análise, escolha Análise de entidade principal: acesso não utilizado.
5. Insira um nome para o analisador.
6. Em Período de rastreamento, insira o número de dias para análise. O analisador avaliará permissões apenas para as entidades do IAM na conta selecionada que existiram durante todo o período de rastreamento. Por exemplo, se você definir um período de rastreamento de 90 dias, apenas as permissões com pelo menos 90 dias serão analisadas, e as descobertas serão geradas se elas não mostrarem nenhum uso durante esse período. Você pode inserir um valor entre 1 e 365 dias.
7. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
8. Em Contas selecionadas, escolha Conta atual.

   > **Note**
   >
   > Se sua conta não for a conta de gerenciamento do AWS Organizations ou uma conta de administrador delegado, você pode criar apenas um analisador com sua conta como a conta selecionada.

9. Opcional. Na seção Excluir perfis e usuários do IAM com tags, é possível especificar pares de valores-chave para perfis e usuários do IAM a serem excluídos da análise de acesso não utilizado. As descobertas não serão geradas para perfis e usuários do IAM excluídos que correspondam aos pares de valores-chave. Para Chave de tag, insira um valor de 1 a 128 caracteres e que não tenha o prefixo `aws:`. Para Valor, é possível inserir um valor de 0 a 256 caracteres de comprimento. Se você não inserir um Valor, a regra será aplicada a todas as entidades principais com a Chave de tag especificada. Escolha Adicionar nova exclusão para adicionar mais pares de valores-chave a serem excluídos.
10. Opcional. Adicione as tags que deseja aplicar ao analisador.
11. Selecione Create analyzer (Criar analisador).

Ao criar um analisador de acessos não utilizados para habilitar o IAM Access Analyzer, uma função vinculada ao serviço chamada `AWSServiceRoleForAccessAnalyzer` será criada em sua conta.

Criação de um analisador de acessos não utilizados com a organização atual

Use o procedimento a seguir para criar um analisador de acessos não utilizados para que uma organização analise centralmente todas as Contas da AWS em uma organização. Para a análise de acessos não utilizados, as descobertas do analisador não mudam com base na região. Não é necessário criar um analisador em cada região em que você tem recursos.

O IAM Access Analyzer cobra pela análise de acessos não utilizados com base no número de usuários e perfis do IAM analisados por mês por analisador. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

> **Note**
>
> Se uma conta-membro for removida da organização, o analisador de acessos não utilizados deixará de gerar novas descobertas e atualizar as descobertas existentes para essa conta após 24 horas. As descobertas associadas à conta do membro que for removida da organização serão removidas permanentemente após 90 dias.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Selecione Create analyzer (Criar analisador).
4. Na seção Análise, escolha Análise de entidade principal: acesso não utilizado.
5. Insira um nome para o analisador.
6. Em Período de rastreamento, insira o número de dias para análise. O analisador avaliará permissões apenas para as entidades do IAM nas contas da organização selecionada que existiram durante todo o período de rastreamento. Por exemplo, se você definir um período de rastreamento de 90 dias, apenas as permissões com pelo menos 90 dias serão analisadas, e as descobertas serão geradas se elas não mostrarem nenhum uso durante esse período. Você pode inserir um valor entre 1 e 365 dias.
7. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
8. Em Contas selecionadas, escolha Organização atual.
9. Opcional. Na seção Excluir Contas da AWS da análise, é possível escolher Contas da AWS em sua organização para excluir da análise de acesso não utilizado. As descobertas não serão geradas para contas excluídas.

a. Para especificar os IDs de contas individuais a serem excluídas, escolha Especificar ID de Conta da AWS e insira os IDs das contas separados por vírgulas no campo ID da Conta da AWS. Escolha Excluir. As contas serão então listadas na tabela Contas da AWS a serem excluídas.

b. Para escolher entre uma lista de contas a serem excluídas em sua organização, escolha Escolher na organização.

   i. É possível pesquisar as contas por nome, e-mail e ID da conta no campo Excluir contas da organização.

   ii. Escolha Hierarquia para visualizar suas contas por unidade organizacional ou escolha Lista para ver uma lista de todas as contas individuais em sua organização.

   iii. Escolha Excluir todas as contas atuais para excluir todas as contas em uma unidade organizacional ou escolha Excluir para excluir contas individuais.

As contas serão então listadas na tabela Contas da AWS a serem excluídas.

> **Note**
>
> As contas excluídas não podem incluir a conta do proprietário do analisador da organização. Quando novas contas são adicionadas à organização, elas não são excluídas da análise, mesmo que você tenha excluído anteriormente todas as contas atuais dentro de uma unidade organizacional. Para obter mais informações sobre a exclusão de contas após a criação de um analisador de acesso não utilizado, consulte Gerenciamento de um analisador de acessos não utilizados do IAM Access Analyzer.

10. Opcional. Na seção Excluir perfis e usuários do IAM com tags, é possível especificar pares de valores-chave para perfis e usuários do IAM a serem excluídos da análise de acesso não utilizado. As descobertas não serão geradas para perfis e usuários do IAM excluídos que correspondam aos pares de valores-chave. Para Chave de tag, insira um valor de 1 a 128 caracteres e que não tenha o prefixo `aws:`. Para Valor, é possível inserir um valor de 0 a 256 caracteres de comprimento. Se você não inserir um Valor, a regra será aplicada a todas as entidades principais com a Chave de tag especificada. Escolha Adicionar nova exclusão para adicionar mais pares de valores-chave a serem excluídos.
11. Opcional. Adicione as tags que deseja aplicar ao analisador.
12. Selecione Create analyzer (Criar analisador).

Ao criar um analisador de acessos não utilizados para habilitar o IAM Access Analyzer, uma função vinculada ao serviço chamada `AWSServiceRoleForAccessAnalyzer` será criada em sua conta.

Gerenciamento de um analisador de acessos não utilizados do IAM Access Analyzer

Use as informações deste tópico para saber como atualizar ou excluir um analisador de acessoas não utilizados existente.

> **Note**
>
> Após você criar ou atualizar um analisador, pode levar algum tempo para as descobertas ficarem disponíveis.

Atualização de um analisador de acessos não utilizados

Use o procedimento a seguir para atualizar um analisador de acessos não utilizados.

O IAM Access Analyzer cobra pela análise de acessos não utilizados com base no número de usuários e perfis do IAM analisados por mês por analisador. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, escolha Configurações do analisador.
3. Na seção Analisadores, escolha o nome do analisador de acesso não utilizado a ser gerenciado.
4. Na guia Exclusão, se o analisador tiver sido criado como o escopo de análise de uma organização, escolha Gerenciar na seção Contas da AWS excluídas.

    a. Para especificar os IDs de contas individuais a serem excluídas, escolha Especificar ID de Conta da AWS e insira os IDs das contas separados por vírgulas no campo ID da Conta da AWS. Escolha Excluir. As contas serão então listadas na tabela Contas da AWS a serem excluídas.

    b. Para escolher entre uma lista de contas a serem excluídas em sua organização, escolha Escolher na organização.

        i. É possível pesquisar as contas por nome, e-mail e ID da conta no campo Excluir contas da organização.

        ii. Escolha Hierarquia para visualizar suas contas por unidade organizacional ou escolha Lista para ver uma lista de todas as contas individuais em sua organização.

iii. Escolha Excluir todas as contas atuais para excluir todas as contas em uma unidade organizacional ou escolha Excluir para excluir contas individuais.

As contas serão então listadas na tabela Contas da AWS a serem excluídas.

c. Para remover contas a serem excluídas, escolha Remover ao lado da conta na tabela Contas da AWS a excluir.

d. Escolha Salvar alterações.

> **Note**
>
> - As contas excluídas não podem incluir a conta do proprietário do analisador da organização.
> - Quando novas contas são adicionadas à organização, elas não são excluídas da análise, mesmo que você tenha excluído anteriormente todas as contas atuais dentro de uma unidade organizacional.
> - Depois de atualizar as exclusões de um analisador, pode levar até dois dias para que a lista de contas excluídas seja atualizada.

5. Na guia Exclusão, escolha Gerenciar na seção Perfis e usuários do IAM excluídos com tags.

   a. É possível especificar pares de valores-chave para perfis e usuários do IAM a serem excluídos da análise de acesso não utilizado. Para Chave de tag, insira um valor de 1 a 128 caracteres e que não tenha o prefixo `aws:`. Para Valor, é possível inserir um valor de 0 a 256 caracteres de comprimento. Se você não inserir um Valor, a regra será aplicada a todas as entidades principais com a Chave de tag especificada.

   b. Escolha Adicionar nova exclusão para adicionar mais pares de valores-chave a serem excluídos.

   c. Para remover pares de valores-chave a serem excluídos, escolha Remover ao lado do par de valores-chave.

   d. Escolha Salvar alterações.

6. Na guia Regras de arquivamento, é possível criar, editar ou excluir regras de arquivamento para o analisador. Para obter mais informações, consulte Regras de arquivamento.

7. Na guia Tags, é possível gerenciar e criar tags para o analisador. Para obter mais informações, consulte Tags para recursos do AWS Identity and Access Management.

Exclusão de um analisador de acessos não utilizados

Use o procedimento a seguir para excluir um analisador de acessos não utilizados. Quando você exclui um analisador, os recursos não são mais monitorados e nenhuma nova descoberta é gerada. Todas as descobertas geradas pelo analisador são excluídas.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acesso, escolha Acesso não utilizado.
3. Em Analisador de acessos, escolha Configurações do analisador.
4. Na seção Analisadores, escolha o nome do analisador de acesso não utilizado a ser excluído.
5. Escolha Excluir analisador.
6. Insira excluir e escolha Excluir para confirmar a exclusão do analisador.

Visualizar o painel de descobertas do IAM Access Analyzer

AWS Identity and Access Management Access AnalyzerO organiza as descobertas de acesso externo, interno e não utilizado em um painel de resumo visual. O painel ajuda você a obter visibilidade do uso de fato de permissões em escala e a identificar as contas e os recursos da AWS que requerem atenção. É possível usar o painel para analisar as descobertas por organização AWS, conta e tipo de descoberta.

Para descobertas de acesso externo e interno:

- O painel destaca a separação entre descobertas de acesso público, descobertas de acesso externo e descobertas de acesso interno.
- O painel fornece um detalhamento das descobertas por tipo de recurso.

Para descobertas de acessos não utilizados:

- O painel destaca as Contas da AWS com mais descobertas de acesso não utilizadas.
- O painel fornece um detalhamento das descobertas por tipo.

Após você criar um analisador de acesso, o IAM Access Analyzer adiciona automaticamente novas descobertas ao painel relevante. Isso permite que você identifique e priorize as áreas com mais problemas de segurança.

Os painéis de resumo oferecem uma visão de alto nível dos problemas de acesso detectados pelo IAM Access Analyzer em todo o seu ambiente da AWS. Em seguida, você pode detalhar as descobertas individuais para investigar mais a fundo e tomar as medidas apropriadas para resolvê-las.

Visualizar o painel de resumo dos analisadores de acesso externo e interno

> **Note**
>
> Após criar ou atualizar um analisador, pode levar algum tempo para que o painel de resumo reflita as atualizações das descobertas.

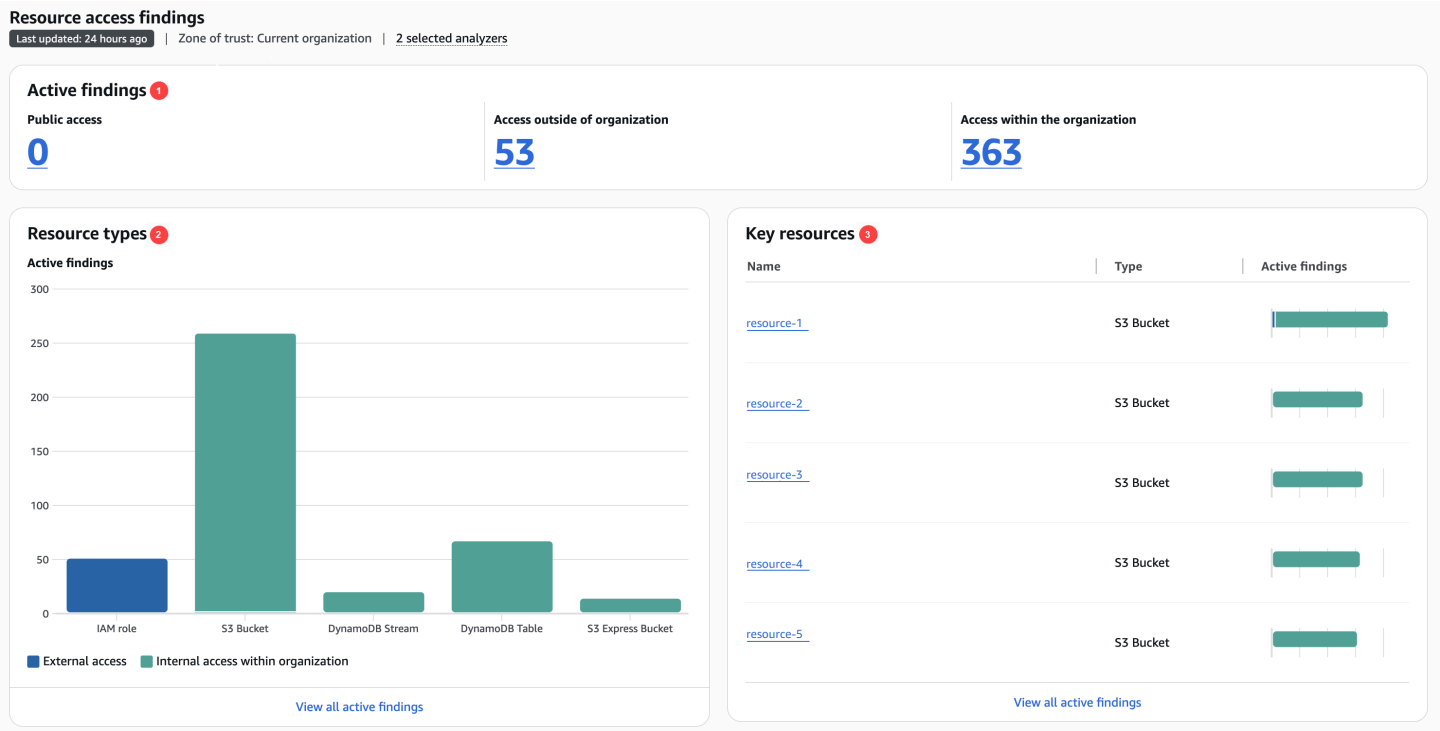
1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. Escolha Analisador de acesso. A janela Resumo será exibida.
3. Escolha Selecionar analisadores.
4. Na janela Selecionar analisadores, escolha Organização ou Conta para a Zona de confiança.

   > **Note**
   >
   > Somente a conta gerencial ou o administrador delegado do AWS Organizations podem escolher Organização como zona de confiança.

5. Escolha analisadores de acesso externo e interno no menu suspenso Analisadores de acesso externo.

   > **Note**
   >
   > Você pode selecionar no máximo um analisador de acesso externo e no máximo um analisador de acesso interno.

6. Selecione Atualizar. Um resumo das descobertas dos analisadores de acesso externo e interno é exibido na seção Descobertas de acesso a recursos.



Na imagem anterior, o painel de descobertas de recursos está visível na página Resumo.

1. A seção Descobertas ativas inclui o número de descobertas ativas para acesso público e o número de descobertas ativas que fornecem acesso fora da conta ou da organização, e o número de descobertas ativas de acesso interno para os analisadores selecionados. Escolha um número para listar todas as descobertas ativas de cada tipo.
2. A seção Tipos de recursos inclui um detalhamento dos tipos de recursos com descobertas ativas para os analisadores selecionados. Escolha Visualizar todas as descobertas ativas para obter uma lista completa das descobertas ativas para os analisadores selecionados.
3. A seção Recursos-chave inclui um resumo dos recursos-chave com descobertas ativas. Essas informações ajudam a priorizar as descobertas para recursos críticos para os negócios. Escolha Visualizar todas as descobertas ativas para obter uma lista completa das descobertas ativas para os analisadores selecionados.

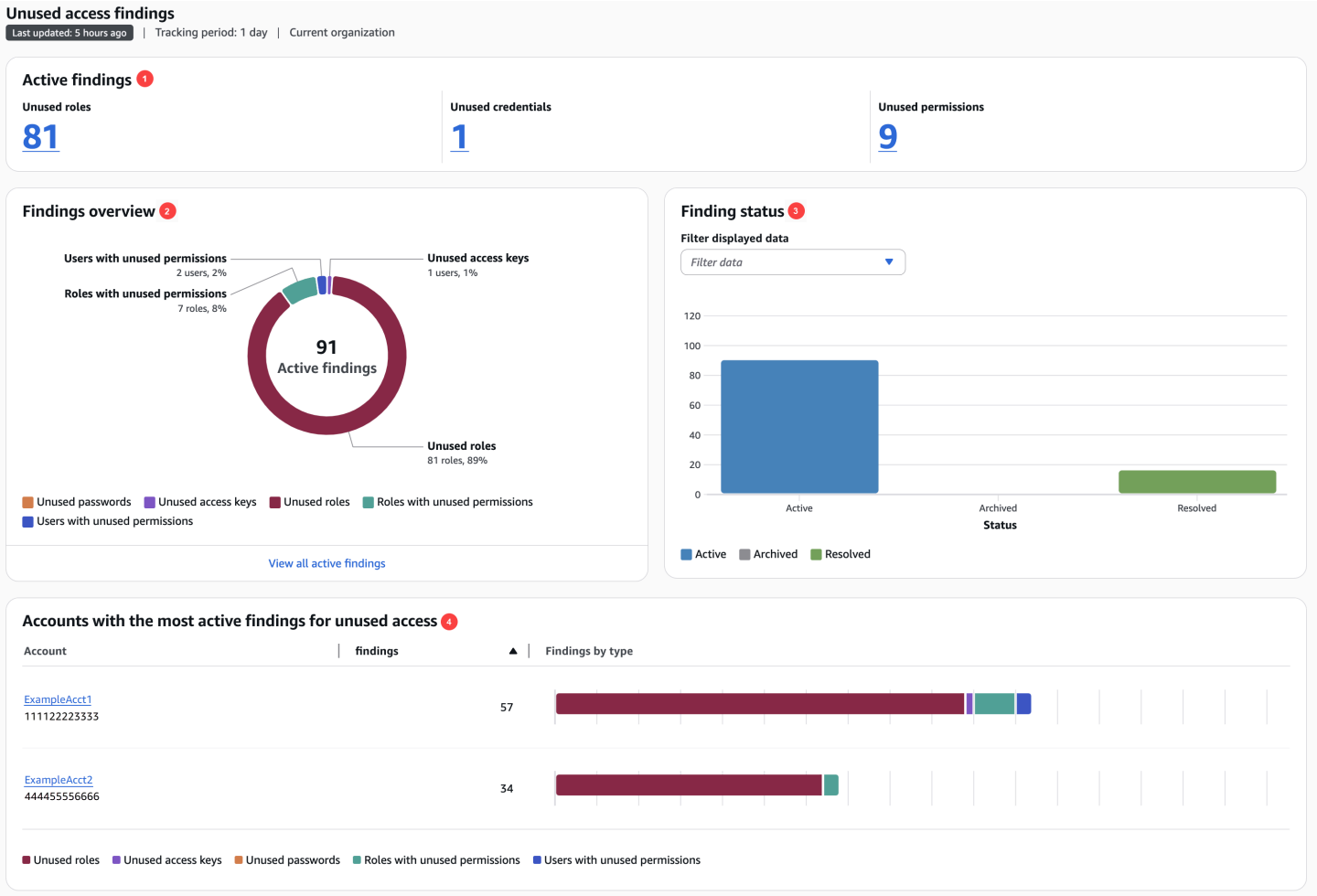
Para visualizar o painel de resumo dos analisadores de acesso não utilizado

> **Note**
>
> Após criar ou atualizar um analisador, com base na quantidade de usuários e funções, pode levar algum tempo para que o painel de resumo reflita as atualizações das descobertas.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Analisador de acesso. A janela Resumo do analisador de acesso é exibida.
3. Escolha Selecionar analisadores.
4. Na janela Selecionar analisadores, escolha Organização ou Conta para a Zona de confiança.

   > **Note**
   >
   > Somente a conta gerencial ou o administrador delegado do AWS Organizations podem escolher Organização como zona de confiança.

5. Escolha um analisador de acesso não utilizado no menu suspenso Analisadores de acesso não utilizado.
6. Escolha Atualizar resumo. Um resumo das descobertas para o analisador de acesso não utilizado selecionado é exibido na seção Descobertas de acesso não utilizado.



Na imagem anterior, o painel de descobertas de acesso não utilizado pode ser visto na página Resumo.

1. A seção Descobertas ativas inclui o número de descobertas ativas para perfis não utilizados, credenciais não utilizadas e permissões não utilizadas em sua conta ou organização. As credenciais não utilizadas incluem tanto as descobertas de chaves de acesso não utilizadas quanto as senhas não utilizadas. As Permissões não utilizadas incluem usuários e perfis com permissões não utilizadas. Escolha um número para listar todas as descobertas ativas de cada tipo.
2. A seção Visão geral das descobertas inclui um detalhamento do tipo de descobertas ativas. Escolha Exibir todas as descobertas ativas para obter uma lista completa das descobertas ativas da conta ou organização do analisador.
3. A seção Status da descoberta inclui um detalhamento do status das descobertas (Ativa, Arquivada e Resolvida) de sua conta ou organização. Você pode selecionar o status das descobertas a serem exibidas no menu suspenso Filtrar dados exibidos.

4. A seção Contas com mais descobertas para acessos não utilizados só é exibida se as contas selecionadas do seu analisador de acessos não utilizados estiverem no nível da organização. Inclui um detalhamento das contas em sua organização com as descobertas mais ativas. Essa não é uma lista completa de todas as contas da sua organização. Seu analisador pode ter descobertas ativas para outras contas não listadas nesta seção.

Revisar descobertas do IAM Access Analyzer

Depois de habilitar o IAM Access Analyzer, a próxima etapa é revisar todas as descobertas para determinar se o acesso identificado na descoberta é intencional ou não. Também é possível revisar as descobertas a fim de determinar descobertas semelhantes para o acesso pretendido e, em seguida, criar uma regra de arquivamento para arquivar essas descobertas automaticamente. Também é possível revisar descobertas resolvidas e arquivadas.

Você deve revisar todas as descobertas da sua conta para determinar se o acesso externo, interno ou não utilizado é esperado e aprovado. Se o acesso identificado na descoberta for esperado, você poderá arquivar a descoberta. Ao arquivar uma descoberta, o status é alterado para Arquivada, e a descoberta é removida da lista de descobertas ativas. A descoberta não é excluída. É possível visualizar as descobertas arquivadas a qualquer momento. Trabalhe em todas as descobertas em sua conta até que não tenha nenhuma descoberta ativa. Após chegar a zero descobertas, você sabe que todas as descobertas geradas com status Ativa são de uma alteração recente no seu ambiente.

> **Tip**
>
> Se você tiver um grande número de descobertas para analisar, use a API do analisador de acesso do IAM ou os comandos da AWS CLI para recuperar e gerenciar programaticamente suas descobertas.

Para revisar as descobertas ativas para todos os tipos de analisadores de acesso

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso). O painel de descobertas é exibido.
3. Escolha Selecionar analisadores.
4. Na janela Selecionar analisadores, escolha somente um analisador de acesso externo e um analisador de acesso interno no menu suspenso Analisadores de acesso a recursos. Escolha um analisador de acesso não utilizado no menu suspenso Analisadores de acesso não utilizado.

5. Escolha Atualizar resumo. Um resumo das descobertas ativas para os analisadores de acesso selecionados é exibido no painel. Escolha um tipo de descoberta nas seções Descobertas de acesso a recursos ou Descobertas de acesso não utilizado para visualizar todas as descobertas ativas do tipo selecionado.

   Para obter mais informações sobre a visualização de descobertas, consulte Visualizar o painel de descobertas do IAM Access Analyzer.

> **Note**
>
> As descobertas serão exibidas somente se você tiver permissão para visualizá-las para o analisador.

Descobertas de acesso externo e interno

> **Note**
>
> O IAM Access Analyzer cobra análise de acesso interno com base no número de recursos monitorados por região por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

1. Em Analisador de acesso, escolha Análise de recursos.
2. Escolha Selecionar analisadores.
3. Na janela Selecionar analisadores, escolha somente um analisador de acesso externo e um analisador de acesso interno no menu suspenso Analisadores de acesso a recursos.
4. Escolha Atualizar resumo.

   A página Análise de recursos exibe os seguintes detalhes sobre os recursos com descobertas ativas para os analisadores de acesso selecionados:

Nome

O nome do recurso com as descobertas ativas.

Tipo

O tipo de recurso.

Conta do proprietário

Essa coluna será exibida somente se você estiver usando uma organização como zona de confiança para um ou mais analisadores selecionados. A conta na organização que possui o recurso relatado na descoberta.

Descobertas ativas

Uma representação visual do número e do tipo de descobertas ativas para o recurso. Passe o mouse sobre o campo para exibir mais informações sobre as descobertas para o recurso.

Acesso público

Indica se alguma das descobertas para o recurso permite acesso público.

Descobertas de acessos não utilizados

> **Note**
>
> O IAM Access Analyzer cobra pela análise de acessos não utilizados com base no número de perfis e usuários do IAM analisados por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

1. Em Analisador de acesso, escolha Acesso não utilizado.
2. Escolha Selecionar analisadores.
3. Na janela Selecionar analisadores, escolha um analisador de acesso não utilizado no menu suspenso Analisadores de acesso não utilizado.
4. Escolha Atualizar resumo.

   A página Acesso não utilizado exibe os seguintes detalhes sobre as entidades do IAM que geraram as descobertas para o analisador de acesso selecionado:

ID* da descoberta

O ID exclusivo atribuído à descoberta. Selecione o ID da descoberta para exibir detalhes adicionais sobre a entidade do IAM que gerou a descoberta.

Tipo de descoberta

O tipo de descoberta de acessos não utilizados: chave de acesso não utilizada, senha não utilizada, permissão não utilizada ou função não utilizada.

Entidade IAM

A entidade IAM relatada na descoberta. Pode ser um usuário ou um perfil do IAM.

Conta da AWS ID da

Essa coluna será exibida somente se você configurar o analisador para todas as Contas da AWS na organização. A Conta da AWS na organização que possui a entidade IAM relatada na descoberta.

Última atualização

A última vez em que a entidade do IAM relatada na descoberta foi atualizada ou quando a entidade foi criada, caso nenhuma atualização tenha sido feita.

Status

O status da descoberta (Ativa, Arquivada ou Resolvida).

Filtrar descobertas do IAM Access Analyzer

A filtragem padrão para uma página de descobertas é exibir todas as descobertas. Para visualizar todas as descobertas, escolha Todas no menu suspenso Status. Para visualizar as descobertas arquivadas, escolha Arquivadas. Para visualizar as descobertas resolvidas, escolha Resolvidas. Ao usar o IAM Access Analyzer pela primeira vez, não haverá descobertas arquivadas.

Use filtros para exibir somente as descobertas que atendem aos critérios de propriedade especificados. Para criar um filtro, selecione a propriedade a ser filtrada, em seguida escolha se a propriedade é igual a ou contém um valor, e depois insira ou escolha um valor de propriedade a ser filtrado.

Para obter uma lista de chaves de filtro que podem ser usadas para criar ou atualizar uma regra de arquivamento, consulte Chaves de filtro do IAM Access Analyzer.

Filtrar recursos com descobertas ativas

Você pode visualizar e filtrar descobertas por recurso para, no máximo, um analisador de acesso externo e um analisador de acesso interno.

Para filtrar recursos com descobertas ativas

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Análise de recursos.
3. Para filtrar por nome do recurso, digite todo ou parte do nome do recurso na caixa de pesquisa.
4. No menu suspenso Filtrar tipo de acesso, escolha o tipo de acesso:
   - Todos os tipos: exibir os recursos com todos os tipos de descobertas de acesso.
   - Acesso público: exibir somente os recursos com descobertas de acesso público.
   - Acesso externo: exibir somente os recursos com descobertas de acesso externo.
   - Acesso interno dentro da organização: exibir somente os recursos com descobertas de acesso interno.
5. No menu suspenso Filtrar tipo de recurso, escolha um tipo de recurso para exibir somente os recursos do tipo selecionado.

Filtrar descobertas de acessos externos

Para filtrar descobertas de acessos externos

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Configurações do analisador e depois escolha o analisador de acesso externo na seção Analisadores.
3. Selecione Visualizar descobertas.
4. Escolha a caixa de pesquisa para exibir uma lista das propriedades disponíveis.
5. Escolha a propriedade que deve ser usada para filtrar as descobertas exibidas.
6. Escolha o valor que deve corresponder à propriedade. Somente as descobertas com esse valor na descoberta serão exibidas.

   Por exemplo, escolha Recurso como a propriedade, escolha Recurso:, digite uma parte do nome ou o nome todo do bucket e pressione Enter. Somente as descobertas do bucket que correspondem aos critérios do filtro serão exibidas. Para criar um filtro que exiba apenas descobertas de recursos que permitem acesso público, escolha a propriedade Acesso público e selecione Acesso público =, e em seguida, escolha Acesso público = verdadeiro.

É possível adicionar outras propriedades para filtrar ainda mais as descobertas exibidas. Ao adicionar outras propriedades, somente as descobertas que correspondem a todas as condições do filtro serão exibidas. A definição de um filtro para exibir descobertas que correspondem a uma propriedade OU a outra propriedade não é compatível. Escolha Limpar filtros para limpar todos os filtros que você definiu e exibir todas as descobertas com o status especificado para seu analisador.

Alguns campos são exibidos somente quando você está visualizando descobertas de um analisador com uma organização como sua zona de confiança.

As seguintes propriedades estão disponíveis para definição de filtros para acesso externo:

- Public access (Acesso público): para filtrar por descobertas de recursos que permitem acesso público, filtre por Public access (Acesso público) e escolha Public access: true (Acesso público: verdadeiro).
- Resource (Recurso): para filtrar por recurso, digite todo ou parte do nome do recurso.
- Resource Type (Tipo de recurso): para filtrar por tipo de recurso, escolha o tipo na lista exibida.
- Conta do proprietário do recurso: use essa propriedade para filtrar pela conta na organização que possui o recurso relatado na descoberta.
- Restrição da política de controle de recursos: use esta propriedade para filtrar pelo tipo de restrição aplicada por uma política de controle de recursos (RCP) do Organizations. Para saber mais, consulte Políticas de controle de recursos (RCPs) no Guia do usuário do AWS Organizations.
  - Falha ao avaliar RCP: houve um erro ao avaliar o RCP.
  - Não aplicável: nenhuma RCP restringe este recurso ou entidade principal. Isso também inclui recursos para os quais ainda não há suporte para as RCPs.
  - Aplicável: o administrador da sua organização definiu restrições por meio de uma RCP que afeta o recurso ou o tipo de recurso. Entre em contato com o administrador da sua organização para obter mais detalhes.
- Conta da AWS: use esta propriedade para filtrar por Conta da AWS com acesso à seção Entidade principal de uma declaração de política. Para filtrar por Conta da AWS, digite todo ou parte do ID de 12 dígitos da Conta da AWS, ou todo ou parte do ARN completo da conta do usuário ou da função externa da AWS que tem acesso aos recursos na conta atual.
- Usuário canônico: para filtrar por usuário canônico, digite o ID do usuário canônico, conforme definido para buckets do Amazon S3. Para saber mais, consulte Identificadores de conta da AWS.
- Federated User (Usuário federado): para filtrar por usuário federado, digite todo ou parte do ARN da identidade federada. Para saber mais, consulte Provedores de identidade e federação.

- ID da descoberta: para filtrar por ID da descoberta, digite todo ou parte do ID da descoberta.
- Erro: para filtrar por tipo de erro, escolha Acesso negado ou Erro interno.
- Principal ARN (ARN da entidade): use esta propriedade para filtrar o ARN da entidade (usuário, função ou grupo do IAM) usado em uma chave de condição aws:PrincipalArn. Para filtrar pelo ARN da entidade principal, digite todo ou parte do ARN do usuário, da função ou do grupo do IAM de uma Conta da AWS externa informada em uma descoberta.
- Principal OrgID (OrgID da entidade): para filtrar por OrgID da entidade, digite todo ou parte do ID da organização associado às entidades principais externas que pertencem à organização da AWS especificada como uma condição na descoberta. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- OrgPaths da entidade principal: para filtrar por caminhos da organização da entidade, digite todo ou parte do ID da organização ou unidade organizacional (UO) da AWS que concede acesso a todas as entidades principais externas que são membros da conta da organização ou UO especificada como uma condição na política. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- Conta de origem: para filtrar por conta de origem, digite todo ou parte do ID da Conta da AWS associado aos recursos, conforme usado em algumas permissões entre serviços na AWS. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- Source ARN (ARN de origem): para filtrar por ARN de origem, digite todo ou parte do ARN especificado como uma condição na descoberta. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- Source IP (IP de origem): para filtrar por IP de origem, digite todo ou parte do endereço IP que permita que entidades externas acessem recursos na conta atual ao usar o endereço IP especificado. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- OrgID de origem: para filtrar por OrgID de origem, digite todo ou parte do ID da organização associado aos recursos, conforme usado em algumas permissões entre serviços na AWS. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- OrgPaths de origem: para filtrar por OrgPaths de origem, digite todo ou parte da unidade organizacional (UO) associada aos recursos, conforme usado em algumas permissões entre serviços na AWS. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- Source VPC (VPC de origem): para filtrar por VPC de origem, digite todo ou parte do ID da VPC que permite que entidades externas acessem recursos na conta atual ao usar a VPC especificada. Para saber mais, consulte Chaves de contexto de condição globais da AWS.

- ARN da VPC de origem: para filtrar por ARN da VPC de origem, digite todo ou parte do ID do ARN da VPC que permite que entidades externas acessem recursos na conta atual ao usar a VPC especificada. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- VPCE de origem: para filtrar por VPCE de origem, digite todo ou parte do ID do endpoint da VPC que permite que entidades externas acessem recursos na conta atual ao usar o endpoint da VPC especificado. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- Conta da VPCE: para filtrar por conta da VPCE, digite todo ou parte do ID de 12 dígitos da Conta da AWS que possui as entidades externas do endpoint da VPC e permite que entidades externas acessem recursos. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- VPCE OrgID: para filtrar por OrgID da VPCE, digite todo ou parte do ID da organização que possui entidades externas do endpoint da VPC e permite que entidades externas acessem recursos. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- OrgPaths da VPCE: para filtrar por OrgPaths da VPCE, digite todo ou parte da unidade organizacional (UO) que possui entidades externas do endpoint da VPC e permite que entidades externas acessem recursos. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- ID do usuário: para filtrar por ID do usuário, digite todo ou parte do ID do usuário do IAM de uma Conta da AWS externa que tenha acesso ao recurso na conta atual. Para saber mais, consulte Chaves de contexto de condição globais da AWS.
- ID de chave do KMS: para filtrar por ID de chave do KMS, digite todo ou parte do ID da chave do KMS especificada como uma condição para acesso ao objeto do Amazon S3 criptografado pelo AWS KMS em sua conta atual.
- Modo de sessão: para filtrar por modo de sessão para buckets de diretório do Amazon S3 (`ReadOnly` ou `ReadWrite`), digite todo ou parte do modo de sessão. Para saber mais, consulte CreateSession na Amazon Simple Storage Service API Reference.
- Google Audience (Público do Google): para filtrar por público do Google, digite todo ou parte do ID da aplicação do Google especificado como uma condição para o acesso à função do IAM em sua conta atual. Para saber mais, consulte Chaves de contexto de condição do IAM e do AWS STS.
- Público do Cognito: para filtrar por público do Cognito, digite todo ou parte do ID do banco de identidades do Amazon Cognito especificado como uma condição para o acesso ao perfil do IAM em sua conta atual. Para saber mais, consulte Chaves de contexto de condição do IAM e do AWS STS.
- Conta do autor da chamada: o ID da Conta da AWS que possui ou contém a entidade que faz a chamada, como um usuário ou perfil do IAM ou um usuário-raiz da conta do IAM. Isso é usado por

serviços que chamam o AWS KMS. Para filtrar por conta do autor da chamada, digite todo ou parte do ID da Conta da AWS.

- Facebook App ID (ID da aplicação do Facebook): para filtrar por ID da aplicação do Facebook, digite todo ou parte do ID da aplicação do Facebook (ou ID do site) especificado como uma condição para conceder acesso à federação Login with Facebook a uma função do IAM em sua conta atual. Para saber mais, consulte a seção id em Chaves de contexto de condição do IAM e do AWS STS.
- Amazon App ID (ID da aplicação da Amazon): para filtrar por ID da aplicação da Amazon, digite todo ou parte do ID da aplicação da Amazon (ou ID do site) especificado como uma condição para conceder acesso à federação Login with Amazon a uma função do IAM em sua conta atual. Para saber mais, consulte a seção id em Chaves de contexto de condição do IAM e do AWS STS.
- Lambda Event Source Token (Token de origem de evento do Lambda): para filtrar por token de origem de evento do Lambda transmitido por integrações com a Alexa, digite toda ou parte da string de token.

Filtrar descobertas de acesso interno

Para filtrar descobertas de acesso interno

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Configurações do analisador e depois escolha o analisador de acesso interno na seção Analisadores.
3. Selecione Visualizar descobertas.
4. Escolha a caixa de pesquisa para exibir uma lista das propriedades disponíveis.
5. Escolha a propriedade que deve ser usada para filtrar as descobertas exibidas.
6. Escolha o valor que deve corresponder à propriedade. Somente as descobertas com esse valor na descoberta serão exibidas.

   Por exemplo, escolha Recurso como a propriedade, escolha Recurso:, digite uma parte do nome ou o nome todo do bucket e pressione Enter. Somente as descobertas do bucket que correspondem aos critérios do filtro serão exibidas.

É possível adicionar outras propriedades para filtrar ainda mais as descobertas exibidas. Ao adicionar outras propriedades, somente as descobertas que correspondem a todas as condições do filtro serão exibidas. A definição de um filtro para exibir descobertas que correspondem a uma

propriedade OU a outra propriedade não é compatível. Escolha Limpar filtros para limpar todos os filtros que você definiu e exibir todas as descobertas com o status especificado para seu analisador.

Alguns campos são exibidos somente quando você está visualizando descobertas de um analisador com uma organização como sua zona de confiança.

Os campos a seguir são exibidos somente quando você está visualizando descobertas para um analisador que está monitorando acesso interno:

- Resource (Recurso): para filtrar por recurso, digite todo ou parte do nome do recurso.
- Resource Type (Tipo de recurso): para filtrar por tipo de recurso, escolha o tipo na lista exibida.
- Conta do proprietário do recurso: use essa propriedade para filtrar pela conta na organização que possui o recurso relatado na descoberta.
- ID da descoberta: para filtrar por ID da descoberta, digite todo ou parte do ID da descoberta.

Filtrar descobertas de acessos não utilizados

Para filtrar descobertas de acessos não utilizados

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Acessos não utilizados e, em seguida, escolha o analisador no menu suspenso Exibir analisador.
3. Escolha a caixa de pesquisa para exibir uma lista das propriedades disponíveis.
4. Escolha a propriedade que deve ser usada para filtrar as descobertas exibidas.
5. Escolha o valor que deve corresponder à propriedade. Somente as descobertas com esse valor na descoberta serão exibidas.

   Por exemplo, escolha Tipo de descobertas como a propriedade, escolha Tipo de descobertas = e depois Tipo de descobertas = perfil não utilizado. Somente as descobertas do tipo Perfil não utilizado serão exibidas.

É possível adicionar outras propriedades para filtrar ainda mais as descobertas exibidas. Ao adicionar outras propriedades, somente as descobertas que correspondem a todas as condições do filtro serão exibidas. A definição de um filtro para exibir descobertas que correspondem a uma propriedade OU a outra propriedade não é compatível. Escolha Limpar filtros para limpar todos os filtros que você definiu e exibir todas as descobertas com o status especificado para seu analisador.

Os campos a seguir são exibidos somente quando você está visualizando descobertas de um analisador que está monitorando acessos não utilizados:

- Tipo de descobertas: para filtrar as descobertas por tipo, filtre por Tipo de descobertas e, em seguida, escolha o tipo de descoberta.
- Resource (Recurso): para filtrar por recurso, digite todo ou parte do nome do recurso.
- Resource Type (Tipo de recurso): para filtrar por tipo de recurso, escolha o tipo na lista exibida.
- Conta do proprietário do recurso: use essa propriedade para filtrar pela conta na organização que possui o recurso relatado na descoberta.
- ID da descoberta: para filtrar por ID da descoberta, digite todo ou parte do ID da descoberta.

Arquivar descobertas do IAM Access Analyzer

Ao obter uma descoberta de acesso a um recurso que é intencional, é possível arquivar as descobertas. Por exemplo, uma descoberta de acesso externo ou interno para um bucket do Amazon S3 acessado por fluxos de trabalho aprovados ou uma descoberta de acesso não utilizado para uma chave de acesso que ainda pode ser necessária. Quando você arquiva uma descoberta, ela é apagada da lista de descobertas ativas. As descobertas arquivadas não são excluídas. É possível filtrar a página Descobertas para exibir as descobertas arquivadas e desarquivá-las a qualquer momento.

Como arquivar descobertas na página Findings (Descobertas)

1. Marque a caixa de seleção ao lado de uma ou mais descobertas que deseja arquivar.
2. Escolha Ações e, em seguida, Arquivar.

   Uma confirmação é exibida na parte superior da tela.

Para arquivar descobertas da página Detalhes de descobertas

1. Selecione o Finding ID (ID da descoberta) que deseja arquivar.
2. Escolha Arquivar.

   Uma confirmação é exibida na parte superior da tela.

Para desarquivar descobertas, repita as etapas anteriores, mas selecione Unarchive (Desarquivar) em vez de Archive (Arquivar). Quando você desarquiva uma descoberta, o status muda para Active (Ativa).

Resolver descobertas do IAM Access Analyzer

Resolver descobertas de recursos

Para resolver descobertas de acesso externo e interno geradas por acesso não pretendido, você deve modificar a instrução da política para remover as permissões de acesso ao recurso identificado.

Para descobertas relacionadas aos buckets do Amazon S3, use o console do Amazon S3 para configurar as permissões no bucket.

Para funções do IAM, use o console do IAM para modificar a política de confiança para a função do IAM listada.

Para outros recursos compatíveis, use o console para modificar as declarações de política que resultaram em uma descoberta gerada.

Após fazer uma alteração para resolver uma descoberta de recurso, como modificar uma política aplicada a um perfil do IAM, o IAM Access Analyzer varrerá o recurso novamente. Se o acesso ao recurso for removido, o status da descoberta será alterado para Resolvido. A descoberta será exibida na lista de descobertas resolvidas em vez da lista de descobertas ativas.

> **Note**
>
> Isso não se aplica às descobertas de Erro. Quando o IAM Access Analyzer não consegue analisar um recurso, ele vai gerar uma descoberta de erro. Se você resolver o problema que impedia o IAM Access Analyzer de analisar o recurso, a descoberta de erro será removida completamente em vez de ser alterada para uma descoberta resolvida. Para obter mais informações, consulte Descobertas de erros do IAM Access Analyzer.

Se as alterações feitas resultarem em acesso externo ou interno ao recurso, mas de maneira diferente, como com outra entidade principal ou para outra permissão, o IAM Access Analyzer resolverá a descoberta original e gerará uma nova descoberta Ativa. Se as alterações feitas resultaram em erros internos ou erros de acesso negado, todas as descobertas ativas sem erro vinculadas ao acesso específico do recurso são resolvidas e uma nova descoberta de erro é gerada.

> **Note**
>
> Para analisadores de acesso externo, depois que uma política é modificada, o IAM Access Analyzer pode levar até 30 minutos para analisar o recurso novamente e atualizar a descoberta.
> Para analisadores de acesso interno, o IAM Access Analyzer pode levar vários alguns minutos ou horas para analisar o recurso novamente e atualizar a descoberta. O IAM Access Analyzer torna a varrer automaticamente todas as políticas a cada 24 horas.
> As descobertas resolvidas são excluídas 90 dias após a última atualização do status da descoberta.

Como resolver descobertas de acessos não utilizados

O IAM Access Analyzer fornece etapas recomendadas para resolver descobertas do analisador de acessos não utilizados com base no tipo de descoberta.

Após fazer uma alteração para resolver uma descoberta de credencial não utilizada, o status da descoberta será alterado para Resolvida na próxima vez em que o analisador de credencial não utilizada for executado. A descoberta não será mais exibida na lista de descobertas ativas e, em vez disso, será exibida na lista de descobertas resolvidas. Se você fizer uma alteração que aborde apenas parcialmente uma descoberta de credencial não utilizada, a descoberta existente será alterada para Resolvida, mas uma nova descoberta será gerada. Por exemplo, se você remover somente algumas das permissões não utilizadas em uma descoberta, mas não todas elas.

O IAM Access Analyzer cobra pela análise de acessos não utilizados com base no número de perfis e usuários do IAM analisados por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Resolvendo descobertas de permissões não utilizadas

Para descobertas de permissões não utilizadas, o IAM Access Analyzer pode recomendar políticas para remover de um usuário ou perfil do IAM e fornecer novas políticas para substituir as políticas de permissões existentes. A recomendação de política não é compatível com os seguintes cenários:

- A descoberta de permissão não utilizada é para um usuário do IAM que está em um grupo de usuários.
- A descoberta de permissão não utilizada é para um perfil do IAM no IAM Identity Center.

- A descoberta de permissão não utilizada tem uma política de permissões existente que inclui o elemento `notAction`.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Acesso não utilizado.
3. Escolha uma descoberta com o Tipo de descoberta definido como Permissões não utilizadas.
4. Na seção Recomendações, se houver políticas listadas na coluna Política recomendada, escolha Visualizar política para visualizar a política existente com a política recomendada para substituir a política existente. Se houver várias políticas recomendadas, você poderá escolher Próxima política e Política anterior para visualizar cada política existente e recomendada.
5. Escolha Baixar JSON para baixar um arquivo .zip com arquivos JSON de todas as políticas recomendadas.
6. Crie e anexe as políticas recomendadas ao usuário ou perfil do IAM. Para obter mais informações, consulte Alterar permissões de um usuário (console) e Modificar uma política de permissões de perfil (console).
7. Remova as políticas listadas na coluna Política de permissões existentes do usuário ou perfil do IAM. Para obter mais informações, consulte como remover as permissões de um usuário (console) e como modificar uma política de permissões de perfil (console).

Resolver descobertas de perfil não utilizado

Para descobertas de perfil não utilizado, o IAM Access Analyzer recomenda excluir o perfil do IAM não utilizado.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Acesso não utilizado.
3. Escolha uma descoberta com o Tipo de descoberta definido como Perfil não utilizado.
4. Na seção Recomendações, analise os detalhes do perfil do IAM.
5. Exclua o perfil do IAM. Para obter mais informações, consulte Exclusão de um perfil do IAM (console).

Como resolver descobertas de chave de acesso não utilizada

Para descobertas de chave de acesso não utilizada, o IAM Access Analyzer recomenda desativar ou excluir a chave de acesso não utilizada.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Acesso não utilizado.
3. Escolha uma descoberta com o Tipo de descoberta definido como Chaves de acesso não utilizadas.
4. Na seção Recomendações, analise os detalhes da chave de acesso.
5. Desative ou exclua a chave de acesso. Para obter mais informações, consulte Gerenciar chaves de acesso (console).

Como resolver descobertas de senha não utilizada

Para descobertas de senha não utilizada, o IAM Access Analyzer recomenda excluir a senha não utilizada para o usuário do IAM.

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Acesso não utilizado.
3. Escolha uma descoberta com o Tipo de descoberta definido como Senha não utilizada.
4. Na seção Recomendações, analise os detalhes do usuário do IAM.
5. Exclua a senha do usuário do IAM. Para obter mais informações, consulte Criar, alterar ou excluir uma senha de usuário do IAM (console).

Descobertas de erros do IAM Access Analyzer

Quando o IAM Access Analyzer analisa recursos, ele normalmente gera descobertas que mostram quem tem acesso aos recursos. Porém, em alguns casos, o analisador pode encontrar problemas que o impedem de concluir a análise. Nessas situações, o IAM Access Analyzer gera descobertas de erros.

As descobertas de erros indicam que o IAM Access Analyzer não conseguiu concluir a análise de um recurso específico ou de um determinado par entidade principal-recurso. Essas descobertas ajudam a identificar os recursos que talvez requeiram atenção para garantir uma análise adequada.

Descobertas de erros de acesso externo

Analisadores de acesso externo, que identificam recursos compartilhados fora de sua conta ou organização, podem gerar dois tipos de descobertas de erros:

- INTERNAL_ERROR: indica que o IAM Access Analyzer encontrou um problema interno ao analisar o recurso. Isso poderia ocorrer devido a limitações do serviço ou a problemas temporários.

```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:iam::941407043048:role/TestAccessAnalyzer",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR",
 "createdAt": "2022-07-14T01:31:43.085000+00:00",
 "resourceType": "AWS::IAM::Role",
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.109000+00:00",
 "id": "4b035c7d-b7d2-40e4-a6c3-9887d1a995df",
 "updatedAt": "2022-07-14T01:31:43.085000+00:00"
}
```

- ACCESS_DENIED: indica que o IAM Access Analyzer não tem as permissões necessárias para analisar o recurso. Isso geralmente acontece quando o acesso ao recurso é negado ao perfil vinculado ao serviço (SLR) do IAM Access Analyzer.

```
{
 "findingDetails": [
  {
   "externalAccessDetails": {}
  }
 ],
 "resource": "arn:aws:kms:us-west-2:941407043048:key/01cae123-
b7f2-4488-9a05-0070a072ea2c",
 "status": "ACTIVE",
 "error": "ACCESS_DENIED",
 "createdAt": "2022-07-14T01:31:43.104000+00:00",
 "resourceType": "AWS::KMS::Key",
```

```
 "findingType": "ExternalAccess",
 "resourceOwnerAccount": "941407043048",
 "analyzedAt": "2025-03-19T06:51:46.090000+00:00",
 "id": "7ef6f04a-9d2c-4038-9cc0-2a5f00a4d8f8",
 "updatedAt": "2022-07-14T01:31:43.104000+00:00"
}
```

Descobertas de erros de acesso interno

Analisadores de acesso interno, que identificam o acesso dentro de sua conta ou organização, podem gerar quatro tipos de descobertas de erros:

- PRINCIPAL_LIMIT_EXCEEDED: gerado quando mais de 3.000 entidades principais têm acesso a um recurso essencial. Esse erro ajuda você a identificar recursos com acesso excessivamente amplo que talvez precise ser limitado.

  Se você fizer alterações no recurso ou nas entidades principais do ambiente que reduzam o número de entidades principais a menos que o limite, o analisador gerará as descobertas normais durante a próxima varredura e a descoberta de erro será marcada como resolvida.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- Erros ao nível do recurso (INTERNAL_ERROR ou ACCESS_DENIED): similares aos erros de acesso externo, indicam que o analisador não conseguiu analisar um determinado recurso devido

a problemas internos ou a problemas de permissão. Quando ocorre um erro ao nível do recurso, o analisador gera uma única descoberta de erro para o recurso em vez das descobertas normais.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

- Erros ao nível da entidade principal (INTERNAL_ERROR ou ACCESS_DENIED): indica que o analisador não conseguiu analisar acesso para um determinado recurso a uma determinada entidade principal. Ao contrário dos erros ao nível do recurso, um recurso pode ter descobertas normais para algumas entidades principais e descobertas de erros para outras entidades principais.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "INTERNAL_ERROR", // can be INTERNAL_ERROR or ACCESS_DENIED
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "findingDetails": [
  {
   "internalAccessDetails": {
    "principal": {
```

```
     "AWS": "arn:aws:iam::111122223333:role/MyRole_1"
    },
    "principalOwnerAccount": "111122223333",
    "principalType": "IAM_ROLE",
    "accessType": "INTRA_ACCOUNT"
   }
  }
 ]
}
```

- PRINCIPAL_ERRORS_LIMIT_EXCEEDED: gerado quando há muitas descobertas de erros ao nível da entidade principal para um único recurso. Essa é uma descoberta de erro ao nível do recurso que pode aparecer junto com as descobertas normais para o mesmo recurso.

```
{
 "id": "efec28fe-b304-412f-af0f-704d0d70c79c",
 "status": "ACTIVE",
 "error": "PRINCIPAL_ERRORS_LIMIT_EXCEEDED",
 "resource": "arn:aws:s3:::critical-data",
 "resourceType": "AWS::S3::Bucket",
 "resourceOwnerAccount": "111122223333",
 "createdAt": "2023-11-30T00:56:56.437000+00:00",
 "analyzedAt": "2024-03-06T04:11:54.406000+00:00",
 "updatedAt": "2023-11-30T00:56:56.437000+00:00",
 "findingType": "InternalAccess",
 "resourceControlPolicyRestriction": "NOT_APPLICABLE",
 "serviceControlPolicyRestriction": "NOT_APPLICABLE",
 "findingDetails": [
  {
   "internalAccessDetails": {}
  }
 ]
}
```

Resolver descobertas de erros

Se você resolver o problema que impedia o IAM Access Analyzer de analisar o recurso, a descoberta de erro será removida completamente em vez de ser alterada para uma descoberta resolvida.

Para resolver as descobertas de erros, considere as seguintes abordagens baseadas no tipo de erro:

- Para erros de ACCESS_DENIED, verifique se o perfil vinculado ao serviço do IAM Access Analyzer tem as permissões necessárias para acessar o recurso.
- Para erros de PRINCIPAL_LIMIT_EXCEEDED, revise as políticas de acesso do recurso e considere restringir o acesso a menos entidades principais.
- Para descobertas de INTERNAL_ERROR, poderá ser necessário aguardar um ciclo de análise subsequente ou entrar em contato com o suporte da AWS se o problema persistir.
- Para PRINCIPAL_ERRORS_LIMIT_EXCEEDED, revise e, se possível, simplifique os padrões de acesso para o recurso afetado.

Depois de fazer alterações para resolver os problemas subjacentes, o IAM Access Analyzer tentará analisar os recursos novamente durante o próximo ciclo de varredura.

Tipos de recursos compatíveis com o IAM Access Analyzer para acesso externo e interno

Para analisadores de acesso externo e interno, o IAM Access Analyzer analisa as políticas baseadas em recurso que são aplicadas aos recursos da AWS na região em que você habilitou o IAM Access Analyzer. Somente analisa as políticas baseadas em recursos. Para obter detalhes sobre como o IAM Access Analyzer gera descobertas para cada tipo de recurso, revise as informações do tipo de recursos.

> **Note**
>
> Os tipos de recursos compatíveis listados são para analisadores de acesso externo e interno. Os analisadores de acesso interno não são compatíveis com todos os tipos de recursos que são compatíveis com os analisadores de acesso externo. Os analisadores de acessos não utilizados oferecem suporte apenas a perfis e usuários do IAM. Para obter mais informações, consulte Como as descobertas do IAM Access Analyzer funcionam.

Tipos de recursos compatíveis para acesso externo

- Buckets do Amazon Simple Storage Service
- Buckets de diretório do Amazon Simple Storage Service
- AWS Identity and Access ManagementFunções do para o

- AWS Key Management ServiceChaves do
- AWS LambdaFunções e camadas do
- Filas do Amazon Simple Queue Service
- AWS Secrets Managersegredos do
- Tópicos do Amazon Simple Notification Service
- Snapshots de volume do Amazon Elastic Block Store
- Snapshots de banco de dados do Amazon Relational Database Service
- Snapshots de cluster de banco de dados do Amazon Relational Database Service
- Repositórios do Amazon Elastic Container Registry
- Sistemas de arquivos do Amazon Elastic File System
- Amazon DynamoDB Streams
- Tabelas do Amazon DynamoDB

Tipos de recursos compatíveis para acesso interno

- Buckets do Amazon Simple Storage Service
- Buckets de diretório do Amazon Simple Storage Service
- Snapshots de banco de dados do Amazon Relational Database Service
- Snapshots de cluster de banco de dados do Amazon Relational Database Service
- Amazon DynamoDB Streams
- Tabelas do Amazon DynamoDB

Buckets do Amazon Simple Storage Service

Quando o IAM Access Analyzer analisa buckets do Amazon S3 para analisadores de acesso externo, ele gera uma descoberta quando uma política de bucket do Amazon S3, uma lista de controle de acesso (ACL) ou um ponto de acesso, incluindo um ponto de acesso multirregional, aplicado a um bucket concede acesso a uma entidade externa. Uma entidade externa é um principal ou outra entidade que pode ser usada para criar um filtro que não esteja em sua zona de confiança. Por exemplo, se uma política de bucket conceder acesso a outra conta ou permitir o acesso público, o IAM Access Analyzer gerará uma descoberta. No entanto, se você habilitar Bloqueio de Acesso Público no bucket, será possível bloquear o acesso no nível da conta ou no nível do bucket.

Para analisadores de acesso interno, o IAM Access Analyzer gera uma descoberta quando uma entidade principal (usuário ou perfil) da sua organização ou conta tem acesso a um bucket do Amazon S3 especificado.

> **Note**
>
> O IAM Access Analyzer não analisa a política de ponto de acesso vinculada aos pontos de acesso entre contas porque o ponto de acesso e a política estão fora da conta do analisador. O IAM Access Analyzer gera uma descoberta pública quando um bucket delega acesso a um ponto de acesso entre contas e o Bloqueio de Acesso Público não está habilitado no bucket ou na conta. Quando você habilita o Bloqueio de Acesso Público, a descoberta pública é resolvida e o IAM Access Analyzer gera uma descoberta entre contas para o ponto de acesso entre contas.

As configurações de Bloqueio de Acesso Público do Amazon S3 substituem as políticas de bucket aplicadas ao bucket. As configurações também substituem as políticas do ponto de acesso aplicadas aos pontos de acesso do bucket. O IAM Access Analyzer analisa as configurações do Bloqueio de Acesso Público no nível do bucket sempre que uma política é alterada. No entanto, ele avalia as configurações do Bloqueio de Acesso Público no nível da conta uma vez a cada seis horas. Isso significa que o IAM Access Analyzer pode não gerar ou resolver uma descoberta para acesso público a um bucket por até seis horas. Por exemplo, se você tem uma política de bucket que permite acesso público, o IAM Access Analyzer gera uma descoberta para esse acesso. Se você habilitar o Bloqueio de Acesso Público para bloquear todo o acesso público ao bucket no nível da conta, o IAM Access Analyzer não resolverá a descoberta para a política de bucket por até seis horas, mesmo que todo o acesso público ao bucket esteja bloqueado. A resolução de descobertas públicas para pontos de acesso entre contas também pode levar até seis horas, depois que você habilitar o Bloqueio de Acesso Público no nível da conta. Alterações em uma política de controle de recursos (RCP) sem uma alteração na política do bucket não acionam uma nova verificação do bucket relatado na descoberta. O IAM Access Analyzer analisa a política nova ou atualizada durante a próxima verificação periódica, que ocorre em até 24 horas.

Para um ponto de acesso de várias regiões, o IAM Access Analyzer usa uma política estabelecida para gerar descobertas. O IAM Access Analyzer avalia as alterações em pontos de acesso de várias regiões uma vez a cada seis horas. Isso significa que o IAM Access Analyzer não gera nem resolve uma descoberta por até seis horas, mesmo que você crie ou exclua um ponto de acesso de várias regiões ou atualize a política para ele.

Buckets de diretório do Amazon Simple Storage Service

Os buckets de diretório do Amazon S3 organizam os dados de forma hierárquica em diretórios, ao contrário da estrutura de armazenamento plana dos buckets de uso geral, que é recomendada para workloads ou aplicações com desempenho crítico. Para analisadores de acesso externo, o IAM Access Analyzer analisa a política de bucket do diretório, incluindo as instruções de condição em uma política, que permite que uma entidade externa acesse um bucket do diretório.

Para analisadores de acesso interno, o IAM Access Analyzer gera uma descoberta quando uma entidade principal (usuário ou perfil) da sua organização ou conta tem acesso a um bucket do diretório do Amazon S3 específico.

Os buckets de diretório do Amazon S3 também oferecem suporte a pontos de acesso, que impõem permissões e controles de rede distintos para todas as solicitações feitas ao bucket de diretórios por meio do ponto de acesso. Cada ponto de acesso pode ter uma política de ponto de acesso personalizada que funciona em conjunto com a política de bucket anexada ao bucket subjacente. Com pontos de acesso para buckets de diretório, você pode restringir o acesso a prefixos específicos, ações de API ou uma nuvem privada virtual (VPC).

> **Note**
>
> O IAM Access Analyzer não analisa a política de ponto de acesso vinculada aos pontos de acesso entre contas porque o ponto de acesso e a política estão fora da conta do analisador. O IAM Access Analyzer gera uma descoberta pública quando um bucket delega acesso a um ponto de acesso entre contas e o Bloqueio de Acesso Público não está habilitado no bucket ou na conta. Quando você habilita o Bloqueio de Acesso Público, a descoberta pública é resolvida e o IAM Access Analyzer gera uma descoberta entre contas para o ponto de acesso entre contas.

Para obter mais informações sobre os buckets de diretório do Amazon S3, consulte Trabalhar com buckets de diretório no Guia do usuário do Amazon Simple Storage Service.

AWS Identity and Access ManagementFunções do para o

Para perfis do IAM, o IAM Access Analyzer analisa políticas de confiança. Em uma política de confiança da função, você define as entidades principais em que confia para assumir a função. Uma política de confiança da função é uma política com base em recurso necessária anexada a uma

função no IAM. O IAM Access Analyzer gera descobertas para funções dentro da zona de confiança que podem ser acessadas por uma entidade externa que esteja fora da sua zona de confiança.

> **Note**
>
> Uma função do IAM é um recurso global. Se uma política de confiança da função conceder acesso a uma entidade externa, o IAM Access Analyzer gerará uma descoberta em cada região habilitada.

AWS Key Management ServiceChaves do

Para AWS KMS keys, o IAM Access Analyzer analisa as políticas de chaves e as concessões aplicadas a uma chave. O IAM Access Analyzer gerará uma descoberta se uma política de chaves ou uma concessão permitir que uma entidade externa acesse a chave. Por exemplo, se você usar a chave de condição kms:CallerAccount em uma instrução de política para permitir o acesso de todos os usuários a uma conta da AWS específica, e especificar uma conta diferente da conta atual (a zona de confiança do analisador atual), o IAM Access Analyzer gerará uma descoberta. Para saber mais sobre as chaves de condições do AWS KMS em instruções de política do IAM, consulte Chaves de condições do AWS KMS.

Quando o IAM Access Analyzer analisa uma chave do KMS, ele lê os metadados da chave, como a política de chaves e a lista de concessões. Se a política de chaves não permitir que a função do IAM Access Analyzer leia os metadados da chave, uma descoberta de erro de Acesso negado será gerada. Por exemplo, se o seguinte exemplo de instrução de política for a única política aplicada a uma chave, isso resultará em uma descoberta de erro de acesso negado no IAM Access Analyzer.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Como essa instrução permite que somente a função chamada Admin da conta da AWS 111122223333 acesse a chave, uma descoberta de erro de acesso negado será gerada, pois o IAM

Access Analyzer não conseguirá analisar a chave completamente. Uma descoberta de erro será exibida em texto vermelho na tabela Findings (Descobertas). A descoberta é semelhante à seguinte:

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Ao criar uma chave do KMS, as permissões concedidas para acessar a chave dependem de como ela foi criada. Se você receber uma descoberta de erro de acesso negado para um recurso de chave, aplique a instrução de política a seguir ao recurso de chave para conceder ao IAM Access Analyzer permissão para acessar a chave.

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Após receber uma descoberta de Acesso negado para um recurso de chave do KMS e resolver a descoberta atualizando a política de chaves, a descoberta será atualizada para um status de Resolved (Resolvida). Se houver instruções de política ou concessões de chave que concedam permissão à chave para uma entidade externa, você poderá ver descobertas adicionais para o recurso de chave.

AWS LambdaFunções e camadas do

Para funções do AWS Lambda, o IAM Access Analyzer analisa políticas, incluindo instruções de condição em uma política, que concedem acesso à função para uma entidade externa. Com o Lambda, você pode anexar políticas exclusivas baseadas em recursos a funções, versões, aliases e camadas. O IAM Access Analyzer relata o acesso externo segundo políticas baseadas em recursos anexadas a funções e camadas. O IAM Access Analyzer não relata o acesso externo segundo políticas baseadas em recursos anexadas a aliases e versões específicas invocadas com um ARN qualificado.

Para obter mais informações, consulte Using resource-based policies for Lambda e Using versions no Guia do desenvolvedor do AWS Lambda.

Filas do Amazon Simple Queue Service

Para filas do Amazon SQS, o IAM Access Analyzer analisa políticas, incluindo instruções de condição em uma política que permitem que uma entidade externa acesse uma fila.

AWS Secrets Managersegredos do

Para segredos do AWS Secrets Manager, o IAM Access Analyzer analisa políticas, incluindo instruções de condição em uma política, que permitem que uma entidade externa acesse um segredo.

Tópicos do Amazon Simple Notification Service

O IAM Access Analyzer analisa políticas baseadas em recursos vinculadas aos tópicos do Amazon SNS, incluindo instruções de condições nas políticas que permitem acesso externo a um tópico. É possível permitir que contas externas realizem ações do Amazon SNS, como assinar e publicar tópicos por meio de uma política baseada em recursos. Um tópico do Amazon SNS será acessível externamente se as entidades principais de uma conta fora da sua zona de confiança puderem realizar operações no tópico. Ao escolher `Everyone` em sua política ao criar um tópico do Amazon SNS, você torna o tópico acessível ao público. `AddPermission` é outra forma de adicionar uma política baseada em recursos a um tópico do Amazon SNS que permite acesso externo.

Snapshots de volume do Amazon Elastic Block Store

Os snapshots de volume do Amazon Elastic Block Store não têm políticas baseadas em recursos. Um snapshot é compartilhado por meio das permissões de compartilhamento do Amazon EBS. Para

snapshots de volume do Amazon EBS, o IAM Access Analyzer analisa listas de controle de acesso que permitem que uma entidade externa acesse um snapshot. Um snapshot de volume do Amazon EBS pode ser compartilhado com contas externas quando criptografado. Um snapshot de volume não criptografado pode ser compartilhado com contas externas e conceder acesso público. As configurações de compartilhamento estão no atributo `CreateVolumePermissions` do snapshot. Quando os clientes visualizam o acesso externo de um snapshot do Amazon EBS, eles podem especificar a chave de criptografia como um indicador de que o snapshot está criptografado, da mesma forma como a versão prévia do IAM Access Analyzer trata os segredos do Secrets Manager.

Snapshots de banco de dados do Amazon Relational Database Service

Os snapshots de banco de dados do Amazon RDS não têm políticas baseadas em recursos. Um snapshot de banco de dados é compartilhado por meio de permissões de banco de dados do Amazon RDS, e somente snapshots de banco de dados manuais podem ser compartilhados. Para analisadores de acesso externo, o IAM Access Analyzer analisa listas de controle de acesso que permitem que uma entidade externa acesse um snapshot do banco de dados do Amazon RDS. Os snapshots de banco de dados não criptografados podem ser públicos. Os snapshots de banco de dados criptografados não podem ser compartilhados publicamente, mas podem ser compartilhados com até 20 outras contas. Para obter mais informações, consulte Criar um snapshot de banco de dados. O IAM Access Analyzer considera a capacidade de exportar um snapshot manual de banco de dados (por exemplo, para um bucket do Amazon S3) como acesso confiável.

Para analisadores de acesso interno, o IAM Access Analyzer gera uma descoberta quando uma entidade principal (usuário ou perfil) da sua organização ou conta tem acesso a um snapshot do banco de dados do Amazon RDS especificado.

> **Note**
>
> O IAM Access Analyzer não identifica o acesso público ou entre contas configurado diretamente no próprio banco de dados. O IAM Access Analyzer identifica apenas descobertas para acesso público ou entre contas configurado no snapshot de banco de dados do Amazon RDS.

Snapshots de cluster de banco de dados do Amazon Relational Database Service

Os snapshots de cluster de banco de dados do Amazon RDS não têm políticas baseadas em recursos. Um snapshot é compartilhado por meio das permissões de cluster de banco de dados

do Amazon RDS. Para analisadores de acesso externo, o IAM Access Analyzer analisa listas de controle de acesso que permitem que uma entidade externa acesse um snapshot do cluster de bancos de dados do Amazon RDS. Os snapshots de cluster não criptografados podem ser públicos. Os snapshots de cluster criptografados não podem ser compartilhados publicamente. Os snapshots de cluster não criptografados e criptografados podem ser compartilhados com até 20 outras contas. Para obter mais informações, consulte Criar um snapshot de cluster de banco de dados. O IAM Access Analyzer considera a capacidade de exportar um snapshot de cluster de banco de dados (por exemplo, para um bucket do Amazon S3) como acesso confiável.

Para analisadores de acesso interno, o IAM Access Analyzer gera uma descoberta quando uma entidade principal (usuário ou perfil) da sua organização ou conta tem acesso a um snapshot do cluster de bancos de dados do Amazon RDS especificado.

> **Note**
>
> As descobertas do IAM Access Analyzer não incluem o monitoramento de qualquer compartilhamento de clones e clusters de banco de dados do Amazon RDS com outra Conta da AWS ou organização usando o AWS Resource Access Manager. O IAM Access Analyzer identifica apenas descobertas para acesso público ou entre contas configurado no snapshot de cluster de banco de dados do Amazon RDS.

Repositórios do Amazon Elastic Container Registry

Para repositórios do Amazon ECR, o IAM Access Analyzer analisa políticas baseadas em recursos, incluindo instruções de condição em uma política que concede a uma entidade externa acesso a um repositório (semelhante a outros tipos de recursos, como tópicos do Amazon SNS e sistemas de arquivos do Amazon EFS). Para repositórios do Amazon ECR, uma entidade principal deve ter permissão para `ecr:GetAuthorizationToken` por meio de uma política baseada em identidade para ser considerada disponível externamente.

Sistemas de arquivos do Amazon Elastic File System

Para sistemas de arquivos do Amazon EFS, o IAM Access Analyzer analisa políticas, incluindo instruções de condição em uma política que permitem que uma entidade externa acesse um sistema de arquivos. Um sistema de arquivos do Amazon EFS será acessível externamente se as entidades principais de uma conta fora da sua zona de confiança puderem realizar operações no sistema de arquivos. O acesso é definido por uma política de sistema de arquivos que usa o IAM e pela forma

como o sistema de arquivos é montado. Por exemplo, montar seu sistema de arquivos Amazon EFS em outra conta é considerado acessível externamente, a menos que essa conta esteja em sua organização e você tenha definido a organização como sua zona de confiança. Se você estiver montando o sistema de arquivos em uma nuvem privada virtual com uma sub-rede pública, o sistema de arquivos estará acessível externamente. Quando você usar o Amazon EFS com o AWS Transfer Family, as solicitações de acesso ao sistema de arquivos recebidas de um servidor do Transfer Family que pertence a uma conta diferente do sistema de arquivos serão bloqueadas se o sistema de arquivos permitir o acesso público.

Amazon DynamoDB Streams

Para analisadores de acesso externo, o IAM Access Analyzer gera uma descoberta se uma política do DynamoDB permitir pelo menos uma ação entre contas que conceda a uma entidade externa acesso a um fluxo do DynamoDB. Para obter mais informações sobre as ações entre contas aceitas pelo DynamoDB, consulte Ações do IAM compatíveis com políticas baseadas em recursos no Guia do desenvolvedor do Amazon DynamoDB.

Para analisadores de acesso interno, o IAM Access Analyzer gera uma descoberta quando uma entidade principal (usuário ou perfil) da sua organização ou conta tem acesso a um fluxo do DynamoDB especificado.

Tabelas do Amazon DynamoDB

Para analisadores de acesso externo, o IAM Access Analyzer gera uma descoberta para uma tabela do DynamoDB se uma política do DynamoDB permitir pelo menos uma ação entre contas que conceda a uma entidade externa acesso a uma tabela ou a um índice do DynamoDB. Para obter mais informações sobre as ações entre contas aceitas pelo DynamoDB, consulte Ações do IAM compatíveis com políticas baseadas em recursos no Guia do desenvolvedor do Amazon DynamoDB.

Para analisadores de acesso interno, o IAM Access Analyzer gera uma descoberta quando uma entidade principal (usuário ou perfil) da sua organização ou conta tem acesso a uma tabela do DynamoDB especificada.

Administrador delegado do IAM Access Analyzer

Se você estiver configurando o AWS Identity and Access Management Access Analyzer IAM Access Analyzer em sua conta de gerenciamento do AWS Organizations, poderá adicionar uma conta-membro na organização como o administrador delegado para gerenciar o IAM Access Analyzer para sua organização. O administrador delegado tem permissões para criar e gerenciar analisadores

dentro da organização. Somente a conta de gerenciamento pode adicionar um administrador delegado.

O administrador delegado do IAM Access Analyzer é uma conta-membro dentro da organização que tem permissões para criar e gerenciar que analisam credenciais em toda a organização. Somente a conta de gerenciamento pode adicionar, remover ou alterar um administrador delegado.

Se você adicionar um administrador delegado, poderá alterar posteriormente para outra conta para o administrador delegado. Ao fazer isso, a conta do administrador delegado anterior perderá as permissões para todos os analisadores que foram criados usando essa conta. Esses analisadores passam para um estado desabilitado e não geram mais descobertas nem atualizam descobertas existentes. As descobertas existentes desses analisadores também não ficam mais acessíveis. Será possível acessá-los novamente no futuro configurando a conta como o administrador delegado. Se você sabe que não usará a mesma conta como administrador delegado, considere excluir os analisadores antes de alterar o administrador delegado. Isso excluirá todas as descobertas geradas. Quando o novo administrador delegado cria outros analisadores, novas instâncias das mesmas descobertas são geradas. Você não perderá nenhuma descoberta. Elas apenas serão geradas para o novo analisador em outra conta. Além disso, você poderá continuar acessando as descobertas para a organização usando a conta de gerenciamento da organização, que também tem permissões de administrador. O novo administrador delegado deve criar outros analisadores para que o IAM Access Analyzer comece a monitorar recursos na organização.

Se o administrador delegado sair da organização da AWS, os privilégios da administração delegada serão removidos da conta. Todos os analisadores da conta com a organização como zona de confiança passam para um estado desabilitado. As descobertas existentes desses analisadores também não ficam mais acessíveis.

Na primeira vez que você configurar analisadores na conta de gerenciamento, é possível escolher a opção Adicionar administrador delegado em Configurações do analisador, no console do IAM Access Analyzer.

> **Note**
>
> O IAM Access Analyzer cobra por analisadores de acessos não utilizados com base no número de usuários e perfis do IAM analisados por analisador por mês. Se você criar um analisador de acessos não utilizados na conta de gerenciamento e na conta de administrador delegado, você será cobrado pelos dois analisadores de acessos não utilizados. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Após alterar o administrador delegado, o novo administrador deverá criar analisadores para começar a monitorar o acesso aos recursos da organização.

Adicionar um administrador delegado ao IAM Access Analyzer

Se você estiver configurando o AWS Identity and Access Management Access Analyzer IAM Access Analyzer em sua conta de gerenciamento do AWS Organizations, poderá adicionar uma conta-membro na organização como o administrador delegado para gerenciar o IAM Access Analyzer para sua organização. O administrador delegado tem permissões para criar e gerenciar analisadores dentro da organização. Somente a conta de gerenciamento pode adicionar um administrador delegado.

Como adicionar um administrador delegado usando o console

1. Faça login no Console AWS usando a conta de gerenciamento da sua organização.
2. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
3. Em Analisador de acessos, selecione Configurações do analisador.
4. Selecione Add delegated administrator (Adicionar administrador delegado).
5. No campo Administrador delegado, insira o número da Conta da AWS de uma conta-membro de organização para torná-la o administrador delegado.

   A conta deve ser membro da organização.
6. Escolha Salvar alterações.

Para adicionar um administrador delegado usando a AWS CLI ou AWS SDKs

Ao criar um analisador para analisar acessos na organização em uma conta de administrador delegado usando a CLI da AWS, a API da AWS (usando os SDKs da AWS) ou o CloudFormation, você deverá usar APIs do AWS Organizations para habilitar o acesso ao serviço para o analisador de acesso do IAM e registrar a conta de membro como administrador delegado.

1. Habilitar o acesso de serviço confiável para o IAM Access Analyzer no AWS Organizations. Consulte Como habilitar ou desabilitar o acesso confiável no Guia do usuário do AWS Organizations.
2. Registre uma conta de membro válida de sua organização da AWS como administrador delegado usando a operação `RegisterDelegatedAdministrator` da API do AWS Organizations ou o comando `register-delegated-administrator` da AWS CLI.

Regras de arquivamento

As regras de arquivamento arquivam automaticamente novas descobertas que atendem aos critérios que você define ao criar a regra. Também é possível aplicar regras de arquivamento retroativamente para arquivar descobertas existentes que atendam aos critérios da regra de arquivamento. Por exemplo, é possível criar uma regra de arquivamento para arquivar automaticamente qualquer descoberta de um bucket do Amazon S3 específico ao qual você concede acesso regularmente. Ou se você conceder acesso a vários recursos para um principal específico, poderá criar uma regra que arquive automaticamente qualquer nova descoberta gerada para o acesso concedido a esse principal. Isso permite que você se concentre somente em descobertas ativas que podem indicar um risco de segurança.

Ao criar uma regra de arquivamento, apenas novas descobertas que correspondam aos critérios da regra serão arquivadas automaticamente. As descobertas existentes não são arquivadas automaticamente. Ao criar uma regra, é possível incluir até 20 valores por critério na regra. Para obter uma lista de chaves de filtro que podem ser usadas para criar ou atualizar uma regra de arquivamento, consulte Chaves de filtro do IAM Access Analyzer.

> **Note**
>
> Ao criar ou editar uma regra de arquivamento, o IAM Access Analyzer não validará os valores incluídos no filtro da regra. Por exemplo, se você adicionar uma regra para corresponder a uma Conta da AWS, o IAM Access Analyzer aceitará qualquer valor no campo, mesmo que não seja um número de conta válido da AWS.

Como criar uma regra de arquivamento

1. Abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. Escolha Analisador de acessos e, em seguida, escolha Configurações do analisador.
3. Na seção Analisadores, escolha o analisador para o qual você deseja criar uma regra de arquivamento.
4. Na guia Regras de arquivamento, escolha Criar regra de arquivamento.
5. Insira um nome para a regra se quiser alterar o nome padrão.
6. Na seção Rule (Regra), em Criteria (Critérios), selecione uma propriedade que deva corresponder à regra.
7. Escolha uma condição para o valor da propriedade, como Contém, É ou Não equivale a.

Os operadores disponíveis dependem da propriedade escolhida.

8. Se preferir, adicione outros valores à propriedade ou adicione outros critérios à regra. Para descobertas de acessos externos, para garantir que sua regra não arquive novas descobertas para acesso público, também é possível incluir o critério Acesso público e defini-lo como falso.

   Para adicionar outro valor a um critério, selecione Add another value (Adicionar outro valor). Para adicionar outro critério à regra, escolha Adicionar critério.

9. Quando terminar de adicionar critérios e valores, escolha Create rule (Criar regra) para aplicar a regra somente a novas descobertas. Escolha Create and archive active findings (Criar e arquivar descobertas ativas) para arquivar descobertas novas e existentes com base nos critérios da regra. Na seção Results (Resultados), é possível revisar a lista de descobertas ativas às quais a regra de arquivamento se aplica.

Por exemplo, para criar uma regra para descobertas de acessos externos que arquive automaticamente qualquer descoberta para buckets do Amazon S3: escolha Tipo de recurso e, em seguida, É como condição. Em seguida, escolha o bucket S3 na lista de Valores.

Para criar uma regra para descobertas de acessos não utilizados que arquive automaticamente qualquer descoberta de uma conta específica: escolha Conta do proprietário do recurso e escolha Equivale a para a condição. Digite o ID da Conta da AWS na caixa de texto Valor.

Continue a definir critérios para personalizar a regra conforme apropriado para o seu ambiente e, em seguida, escolha Criar regra.

Se você criar uma nova regra e adicionar vários critérios, poderá remover um único critério da regra ao selecionar Remover este critério. É possível remover um valor adicionado para um critério selecionando Remove value (Remover valor).

Como editar uma regra de arquivamento

1. Escolha o nome da regra a ser editada na coluna Nome.

   Você pode editar apenas uma regra de arquivamento por vez.

2. Adicione novos critérios ou remova os critérios ou valores existentes para cada critério.
3. Escolha Save changes (Salvar alterações) para aplicar a regra somente a novas descobertas. Escolha Save and archive active findings (Salvar e arquivar descobertas ativas) para arquivar descobertas novas e existentes com base nos critérios da regra.

Como excluir uma regra de arquivamento

1. Marque a caixa de seleção para as regras que deseja excluir.
2. Escolha Excluir.
3. Digite **`delete`** na caixa de diálogo de confirmação Delete archive rule (Excluir regra de arquivamento) e selecione Delete (Excluir).

As regras são excluídas apenas do analisador na região atual. É necessário excluir regras de arquivamento separadamente para cada analisador criado em outras regiões.

Monitoramento do AWS Identity and Access Management Access Analyzer com o Amazon EventBridge

Use as informações neste tópico para aprender como monitorar as descobertas do IAM Access Analyzer e acessar as pré-visualizações com o Amazon EventBridge. O EventBridge é a nova versão do Amazon CloudWatch Events.

Eventos de descobertas

O IAM Access Analyzer envia um evento ao EventBridge para cada descoberta gerada, para uma alteração no status de uma descoberta existente e quando uma descoberta é excluída. Para receber descobertas e notificações sobre descobertas, é necessário criar uma regra de evento no Amazon EventBridge. Ao criar uma regra de evento, também é possível especificar uma ação de destino a ser acionada com base na regra. Por exemplo, você pode criar uma regra de evento que acione um tópico do Amazon SNS quando um evento de uma nova descoberta for recebido do IAM Access Analyzer. Detalhes sobre a política de controle de recursos (RCP) estão disponíveis na seção de detalhes do evento.

Acessar eventos de pré-visualização

O IAM Access Analyzer envia um evento ao EventBridge para cada pré-visualização de acesso e alteração de seu status. Isso inclui um evento quando a pré-visualização de acesso é criada pela primeira vez (status Creating [Criando]), quando a pré-visualização de acesso é concluída (status Completed [Concluída]) ou quando a criação da pré-visualização de acesso falha (status Failed [Falha]). Para receber notificações sobre pré-visualizações de acesso, você deve criar uma regra de evento no EventBridge. Ao criar uma regra de evento, é possível especificar uma ação de destino a ser acionada com base na regra. Por exemplo, você pode criar uma regra de evento que acione um

tópico do Amazon SNS quando um evento de uma prévia de acesso concluído for recebido do IAM Access Analyzer.

Frequência das notificações de eventos

O IAM Access Analyzer envia eventos para novas descobertas e descobertas com atualizações de status ao EventBridge aproximadamente uma hora após o momento no qual o evento ocorre na conta. O IAM Access Analyzer também envia eventos ao EventBridge quando uma descoberta resolvida é excluída porque o período de retenção expirou. Para descobertas que são excluídas porque o analisador que as gerou foi excluído, o evento é enviado ao EventBridge aproximadamente 24 horas após a exclusão do analisador. Quando uma descoberta é excluída, o status da descoberta não é alterado. Em vez disso, o atributo `isDeleted` é definido como `true`. O IAM Access Analyzer também envia eventos para pré-visualizações de acesso recém-criadas e alterações de status de pré-visualização de acesso ao EventBridge.

Exemplos de eventos de descobertas de acessos externos

Veja a seguir um exemplo de evento de descoberta de credencial externa do IAM Access Analyzer enviado ao EventBridge. O `id` listado é o ID do evento no EventBridge. Para saber mais, consulte Eventos e padrões de evento no EventBridge.

No objeto `detail`, os valores dos atributos `accountId` e `region` fazem referência à conta e à região informada na descoberta. O atributo `isDeleted` indica se o evento era da descoberta que está sendo excluída. O `id` é o ID da descoberta. A matriz `resources` é um singleton com o ARN do analisador que gerou a descoberta.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
```

```
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

O IAM Access Analyzer também envia eventos ao EventBridge para descobertas de erro. Uma descoberta de erro é uma descoberta gerada quando o IAM Access Analyzer não consegue analisar o recurso. Os eventos de descobertas de erro incluem um atributo `error`, conforme mostrado no exemplo a seguir.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
```

```
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

Exemplos de eventos de descobertas de acesso interno

O exemplo a seguir é de evento de descoberta de acesso interno do IAM Access Analyzer enviado ao EventBridge. O `id` listado é o ID do evento no EventBridge. Para saber mais, consulte Eventos e padrões de evento no EventBridge.

No objeto `detail`, os valores dos atributos `accountId` e `principalOwnerAccount` referenciam a conta da entidade principal informada na descoberta. O atributo `isDeleted` indica se o evento era da descoberta que está sendo excluída. O `id` é o ID da descoberta. O `resource` é o ARN do analisador que gerou a descoberta.

```
{
    "version": "0",
    "id": "b45c3678-c278-b593-6121-c155259ce1b5",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2025-04-08T19:42:49Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:access-analyzer:us-east-1:111122223333:analyzer/testAnalyzer"
    ],
    "detail": {
        "accessType": "INTRA_ACCOUNT",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2025-04-08T03:18:43.509465073Z",
        "condition": {},
        "createdAt": "2025-04-07T21:33:49.914099224Z",
        "id": "11111111-2222-4444-aaaa-333333333333",
        "isDeleted": false,
        "findingType": "InternalAccess",
```

```
        "principal": {
            "AWS": "arn:aws:iam::111122223333:role/MyRole_6"
        },
        "principalOwnerAccount": "111122223333",
        "principalType": "IAM_ROLE",
        "resource": "arn:aws:s3:::critical-data",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "accountId": "111122223333",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "status": "ACTIVE",
        "updatedAt": "2025-04-08T03:22:12.654688231Z",
        "version": "1.0"
    }
}
```

O IAM Access Analyzer também envia eventos ao EventBridge para descobertas de erro. Uma descoberta de erro é uma descoberta gerada quando o IAM Access Analyzer não consegue analisar o recurso. Os eventos de descobertas de erro incluem um atributo `error`, conforme mostrado no exemplo a seguir.

```
{
    "version": "0",
    "id": "5a94b99b-e87d-a6a7-58c7-f47871532860",
    "detail-type": "Internal Access Finding",
    "source": "aws.access-analyzer-test",
    "account": "444455556666",
    "time": "2025-05-07T11:57:54Z",
    "region": "us-west-2",
    "resources": ["arn:aws:access-analyzer-beta:us-west-2:444455556666:analyzer/
example-analyzer"],
    "detail": {
        "analyzedAt": "2025-03-24T19:58:52.512329448Z",
        "createdAt": "2025-03-22T03:30:46.920200692Z",
        "id": "ef573afd-12a5-4095-87a6-bf2f25109895",
        "isDeleted": false,
        "findingType": "InternalAccess",
        "resource": "arn:aws:s3:::test-entity-88",
        "accountId": "111122223333",
        "resourceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "resourceType": "AWS::S3::Bucket",
        "serviceControlPolicyRestrictionType": "NOT_APPLICABLE",
        "error": "ACCESS_DENIED", // can be INTERNAL_ERROR and ACCESS_DENIED
```

```
        "status": "ACTIVE",
        "updatedAt": "2025-03-24T20:09:39.176075014Z",
        "version": "1.0"
    }
}
```

Exemplo de eventos relacionados a descobertas de acessos não utilizados

Veja a seguir um exemplo de evento de descoberta de credencial não utilizada do IAM Access Analyzer enviado ao EventBridge. O `id` listado é o ID do evento no EventBridge. Para saber mais, consulte Eventos e padrões de evento no EventBridge.

No objeto `detail`, os valores dos atributos `accountId` e `region` fazem referência à conta e à região informada na descoberta. O atributo `isDeleted` indica se o evento era da descoberta que está sendo excluída. O `id` é o ID da descoberta.

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
```

```
        }
    }
```

O IAM Access Analyzer também envia eventos ao EventBridge para descobertas de erro. Uma descoberta de erro é uma descoberta gerada quando o IAM Access Analyzer não consegue analisar o recurso. Os eventos de descobertas de erro incluem um atributo `error`, conforme mostrado no exemplo a seguir.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

Exemplo de eventos de pré-visualização de acesso

O exemplo a seguir mostra dados do primeiro evento enviado ao EventBridge ao criar uma pré-visualização de acesso. A matriz `resources` é um singleton com o ARN do analisador ao qual a pré-visualização de acesso está associada. No objeto `detail`, o `id` refere-se ao ID de pré-

visualização de acesso e `configuredResources` refere-se ao recurso para o qual a pré-visualização de acesso foi criada. O `status` é `Creating` e refere-se ao status de pré-visualização de acesso. O `previousStatus` não é especificado porque a pré-visualização de acesso acabou de ser criada.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

O exemplo a seguir mostra dados de um evento enviado ao EventBridge para uma pré-visualização de acesso com uma alteração de status de `Creating` para `Completed`. No objeto de detalhes, o `id` refere-se ao ID de pré-visualização de acesso. O `status` e `previousStatus` referem-se ao status de pré-visualização de acesso, em que o status anterior era `Creating` e o status atual é `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
```

```
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

O exemplo a seguir mostra dados de um evento enviado ao EventBridge para uma pré-visualização de acesso com uma alteração de status de `Creating` para `Failed`. No objeto `detail`, o `id` refere-se ao ID de pré-visualização de acesso. O `status` e `previousStatus` referem-se ao status de pré-visualização de acesso, em que o status anterior era `Creating` e o status atual é `Failed`. O campo `statusReason` fornece o código de motivo indicando que a pré-visualização de acesso falhou devido a uma configuração de recurso inválida.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
```

```
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Criar uma regra de evento usando o console

O procedimento a seguir descreve como criar uma regra de evento usando o console.

1. Abra o console do Amazon EventBridge em https://console.aws.amazon.com/events/.
2. Usando os valores a seguir, crie uma regra EventBridge que monitore os eventos de descobertas ou acesse os eventos de visualização:

   - Em Tipo de regra, escolha Regra com um padrão de evento.
   - Em Event source (Origem do evento), escolha Other (Outra).
   - Em Event pattern (Padrão de evento), escolha Custom patterns (JSON editor), (Padrões personalizados [editor JSON]) e cole um dos seguintes exemplos de padrão de evento na área de texto:
     - Para criar uma regra baseada em um evento do IAM Access Analyzer, use o seguinte exemplo de padrão:

       ```
       {
         "source": [
           "aws.access-analyzer"
         ]
       }
       ```

     - Para criar uma regra baseada em um evento de descoberta de acesso externo, acesso interno ou acesso não utilizado, use o seguinte exemplo de padrão:

       ```
       {
         "source": [
           "aws.access-analyzer"
         ],
         "detail-type": [
           "Access Analyzer Finding",
           "Internal Access Finding",
       ```

```
    "Unused Access Finding for IAM entities"
  ]
}
```

- Para criar uma regra baseada somente em um evento de descobertas de acesso externo, use o seguinte exemplo de padrão:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
  ]
}
```

- Para criar uma regra baseada somente em um evento de descoberta de acesso interno, use o seguinte exemplo de padrão:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Internal Access Finding"
  ]
}
```

- Para criar uma regra baseada somente em um evento de descobertas não utilizado, use o seguinte exemplo de padrão:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

- Para criar uma regra baseada em um evento de visualização de acesso, use o seguinte exemplo de padrão:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- Em Tipos de destino, escolha Serviço da AWS e, em Selecionar um destino, escolha um destino, como um tópico do Amazon SNS ou um perfil do AWS Lambda. O destino é acionado quando é recebido um evento que corresponde ao padrão de evento definido na regra.

Para saber mais sobre a criação de regras, consulte Creating Amazon EventBridge rules that react to events (Criar regras do Amazon EventBridge que reajam a eventos) no Amazon EventBridge User Guide (Guia do usuário do Amazon EventBridge).

Criar uma regra de evento usando a CLI

1. Use o seguinte para criar uma regra para o Amazon EventBridge usando a AWS CLI. Substitua o nome da regra *TestRule* pelo nome da sua regra.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"]}"
```

2. É possível personalizar a regra para acionar ações de destino somente para um subconjunto de descobertas geradas, como descobertas com atributos específicos. O exemplo a seguir demonstra como criar uma regra que aciona uma ação de destino somente para descobertas com um status de Active (Ativa).

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":[\"ACTIVE\"]}}"
```

O exemplo a seguir demonstra como criar uma regra que acione uma ação de destino apenas para pré-visualizações de acesso com um status de `Creating` para `Completed`.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Para definir uma função Lambda com um destino para a regra criada, use o exemplo de comando a seguir. Substitua a região e o nome da função no ARN conforme apropriado para o seu ambiente.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Adicione as permissões necessárias para invocar o destino da regra. O exemplo a seguir demonstra como conceder permissões a uma função Lambda, seguindo os exemplos anteriores.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Integrar o IAM Access Analyzer com o AWS Security Hub CSPM

O AWS Security Hub CSPM fornece uma visualização abrangente de seu estado de segurança na AWS. Ele ajuda você a avaliar seu ambiente em relação aos padrões e práticas recomendadas do setor de segurança. O Security Hub CSPM coleta dados de segurança de Contas da AWS, serviços e produtos de parceiros terceirizados compatíveis. Em seguida, ele analisa suas tendências de segurança e identifica os problemas de segurança de maior prioridade.

Ao integrar o IAM Access Analyzer com o Security Hub CSPM, você pode enviar descobertas do IAM Access Analyzer para o Security Hub CSPM. O Security Hub CSPM pode então incluir tais descobertas na análise feita sobre a seu procedimento de segurança geral.

Sumário

- Como o IAM Access Analyzer envia descobertas para o Security Hub CSPM
  - Tipos de descobertas que o IAM Access Analyzer envia
  - Latência para enviar descobertas
  - Nova tentativa quando o CSPM do Security Hub não está disponível
  - Atualização das descobertas existentes no CSPM do Security Hub
- Visualizar descobertas do IAM Access Analyzer no Security Hub CSPM

  - Interpretação de nomes de descobertas do IAM Access Analyzer no Security Hub CSPM
- Descoberta típica do IAM Access Analyzer
- Habilitar e configurar a integração
- Como parar de enviar descobertas

Como o IAM Access Analyzer envia descobertas para o Security Hub CSPM

No CSPM do Security Hub, os problemas de segurança são rastreados como descobertas. Algumas descobertas provêm de problemas que são detectados por outros Serviços da AWS ou por parceiros terceirizados. O CSPM do Security Hub também tem um conjunto de regras que ele usa para detectar problemas de segurança e gerar descobertas.

O CSPM do Security Hub fornece ferramentas para gerenciar descobertas em todas essas origens. É possível visualizar e filtrar listas de descobertas e visualizar informações detalhadas sobre cada uma delas. Para obter mais informações, consulte Visualizar descobertas no Guia do usuário do AWS Security Hub. Também é possível acompanhar o status das investigações sobre as descobertas. Para obter mais informações, consulte Tomar medidas sobre descobertas no Manual do usuário do AWS Security Hub.

Todas as descobertas no CSPM do Security Hub usam um formato JSON padrão chamado Formato de Descobertas de Segurança da AWS (ASFF). O ASFF inclui detalhes sobre a origem do problema, os recursos afetados e o status atual da descoberta. Para obter mais informações, consulte AWS Security Finding Format (ASFF) no Manual do usuário do AWS Security Hub.

O AWS Identity and Access Management Access Analyzer é um dos Serviços da AWS que enviam descobertas ao Security Hub CSPM. Para acessos não utilizados, o IAM Access Analyzer detecta o acesso não utilizado concedido a usuários ou perfis do IAM e gera uma descoberta para cada um deles. O IAM Access Analyzer então envia essas descobertas ao Security Hub CSPM.

Para acesso externo, o IAM Access Analyzer detecta declarações de política que permitem acesso público ou acesso entre contas a entidades principais externas em recursos compatíveis em sua organização ou conta. O IAM Access Analyzer gera uma descoberta para acesso público e a envia para o Security Hub CSPM. Para acesso entre contas, o IAM Access Analyzer envia uma única descoberta de uma entidade principal externa por vez ao Security Hub CSPM. Se houver várias descobertas entre contas no IAM Access Analyzer, você deve resolver a descoberta do Security Hub CSPM da única entidade principal externa antes que o IAM Access Analyzer forneça a próxima descoberta entre contas. Para obter uma lista completa de entidades principais externas com acesso

entre contas fora da zona de confiança do analisador, você deve visualizar as descobertas no IAM Access Analyzer. Detalhes sobre a política de controle de recursos (RCP) estão disponíveis na seção de detalhes do recurso.

Tipos de descobertas que o IAM Access Analyzer envia

O IAM Access Analyzer envia as descobertas para o Security Hub CSPM usando o AWS Security Finding Format (ASFF). No ASFF, o campo `Types` fornece o tipo de descoberta. As descobertas do IAM Access Analyzer podem ter os seguintes valores para `Types`.

- Descobertas de acessos externos: efeitos/exposição de dados/acesso externo concedido
- Descobertas de acessos externos: verificações de software e configuração/Práticas recomendadas de segurança da AWS/Acesso externo concedido
- Descobertas de acessos não utilizadas: verificações de software e configuração/Práticas recomendadas de segurança da AWS/Permissão não utilizada
- Descobertas de acessos não utilizados: verificações de software e configuração/Práticas recomendadas de segurança da AWS/Perfil do IAM não utilizado
- Descobertas de acessos não utilizados: verificações de software e configuração/Práticas recomendadas de segurança da AWS/Senha de usuário do IAM não utilizada
- Descobertas de acessos não utilizados: verificações de software e configuração/Práticas recomendadas de segurança da AWS/Chave de acesso de usuário do IAM não utilizada

Latência para enviar descobertas

Quando o IAM Access Analyzer cria um nova descoberta, ela geralmente é enviada para o Security Hub CSPM em até 30 minutos. No entanto, há casos raros em que o IAM Access Analyzer pode não ser notificado sobre uma alteração na política. Por exemplo:

- As alterações nas configurações de bloqueio de acesso público no nível da conta no Amazon S3 podem levar até 12 horas para serem refletidas no IAM Access Analyzer.
- Alterações em uma política de controle de recursos (RCP) sem uma alteração na política baseada no recurso não acionam uma nova verificação do bucket relatado na descoberta. O IAM Access Analyzer analisa a política nova ou atualizada durante a próxima verificação periódica, que ocorre em até 24 horas.
- Se houver um problema de entrega com a entrega de logs do AWS CloudTrail, uma alteração de política pode não acionar uma nova varredura do recurso que foi relatado na descoberta.

Nessas situações, o IAM Access Analyzer analisa a política nova ou atualizada durante a próxima verificação periódica.

Nova tentativa quando o CSPM do Security Hub não está disponível

Se o Security Hub CSPM não estiver disponível, o IAM Access Analyzer tentará enviar as descobertas em intervalos regulares.

Atualização das descobertas existentes no CSPM do Security Hub

Depois de enviar uma descoberta para o Security Hub CSPM, o IAM Access Analyzer continua a enviar atualizações para refletir quaisquer observações adicionais da atividade de descobertas para o Security Hub CSPM. Essas atualizações são refletidas dentro da mesma descoberta.

Para descobertas de acesso externo, o IAM Access Analyzer as agrupa por recurso. No Security Hub CSPM, a descoberta de um recurso permanecerá ativa se pelo menos uma das descobertas desse recurso estiver ativa no IAM Access Analyzer. Se todas as descobertas no IAM Access Analyzer para um recurso forem arquivadas ou resolvidas, então a descoberta do Security Hub CSPM também será arquivada. A descoberta do Security Hub CSPM é atualizada quando o acesso de políticas é alterado entre público e entre contas. Essa atualização pode incluir alterações no tipo, título, descrição e gravidade da descoberta.

Para descobertas de acesso não utilizadas, o IAM Access Analyzer não as agrupa por recurso. Dessa forma, se uma descoberta de acesso não utilizado for resolvida no IAM Access Analyzer, a descoberta do Security Hub CSPM correspondente também será resolvida. A descoberta do Security Hub CSPM é atualizada quando você atualiza o perfil ou o usuário do IAM que gerou a descoberta de acesso não utilizado.

Visualizar descobertas do IAM Access Analyzer no Security Hub CSPM

Para visualizar suas descobertas do IAM Access Analyzer no Security Hub CSPM, escolha Ver descobertas na seção AWS: IAM Access Analyzer da página de resumo. Como alternativa, é possível escolher Findings (Descobertas) no painel de navegação. Em seguida, você pode filtrar as descobertas para exibir somente as descobertas do AWS Identity and Access Management Access Analyzer escolhendo o campo Product name: (Nome do produto:) com um valor de **`IAM Access Analyzer`**.

Interpretação de nomes de descobertas do IAM Access Analyzer no Security Hub CSPM

O AWS Identity and Access Management Access Analyzer envia descobertas para o Security Hub CSPM usando o AWS Security Finding Format (ASFF). No ASFF, o campo Types (Tipos) fornece

o tipo de descoberta. Os tipos do ASFF utilizam um esquema de nomenclatura diferente do AWS Identity and Access Management Access Analyzer. A tabela a seguir inclui detalhes sobre todos os tipos do ASFF associados às descobertas do AWS Identity and Access Management Access Analyzer, conforme aparecem no Security Hub CSPM.

Tipo de descoberta do ASFF	Título da descoberta do Security Hub CSPM	Descrição
Efeitos/Exposição de dados/Acesso externo concedido	O <resource ARN> permite acesso público	Uma política baseada em recursos anexada ao recurso permite o acesso público no recurso a todas as entidades principais externas.
Verificações de software e configuração/Melhores práticas de segurança daAWS/Acesso externo concedido	O <resource ARN> permite o acesso entre contas	Uma política baseada em recursos anexada ao recurso permite o acesso entre contas a entidades principai s externas fora da zona de confiança do analisador.
Verificações de software e configuração/Práticas recomendadas de segurança da AWS/Permissão não utilizada	O <resource ARN> contém permissões não utilizadas	Um usuário ou perfil contém permissões de serviço e ação não utilizadas.
Verificações de software e configuração/Práticas recomendadas de segurança da AWS/Perfis do IAM não utilizados	O <resource ARN> contém um perfil do IAM não utilizado	Um usuário ou função contém um perfil do IAM não utilizado.
Verificações de software e configuração/Práticas recomendadas de segurança da AWS/Senha de usuário do IAM não utilizada	<resource ARN> contém senha de usuário do IAM não utilizada	Um usuário ou função contém uma senha de usuário do IAM não utilizada.

Tipo de descoberta do ASFF	Título da descoberta do Security Hub CSPM	Descrição
Verificações de software e configuração/Práticas recomendadas de segurança da AWS/Chave de acesso de usuário do IAM não utilizada	<resource ARN> contém chave de acesso de usuário do IAM não utilizada	Um usuário ou função contém uma chave de acesso de usuário do IAM não utilizada.

Descoberta típica do IAM Access Analyzer

O IAM Access Analyzer envia as descobertas para o Security Hub CSPM usando o AWS Security Finding Format (ASFF).

Veja aqui um exemplo de uma descoberta típica do IAM Access Analyzer para descobertas de acessos externos.

```
{
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::amzn-s3-demo-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-account
 access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::amzn-s3-demo-bucket/ allows cross-
account access from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
```

```
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
}
```

Veja aqui um exemplo de uma descoberta típica do IAM Access Analyzer para descobertas de acessos não utilizados.

```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
```

```
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
```

```
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```

Habilitar e configurar a integração

Para usar a integração com o Security Hub CSPM, você deve habilitá-lo. Para obter informações sobre como habilitar o Security Hub CSPM, consulte Configurar o Security Hub no Guia do usuário do AWS Security Hub.

Ao habilitar tanto o IAM Access Analyzer quanto o Security Hub CSPM, a integração é habilitada automaticamente. O IAM Access Analyzer começa imediatamente a enviar descobertas para o Security Hub CSPM.

Como parar de enviar descobertas

Para interromper o envio de descobertas ao CSPM do Security Hub, você poderá usar o console ou a API do CSPM do Security Hub.

Consulte Desabilitar e habilitar o fluxo de descobertas de uma integração (console) ou Desabilitar o fluxo de descobertas de uma integração (API do Security Hub, AWS CLI) no Guia do usuário do AWS Security Hub.

Registrar em log chamadas de API do IAM Access Analyzer com o AWS CloudTrail

O IAM Access Analyzer é integrado ao AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, uma função ou um produto da AWS no IAM Access Analyzer. O CloudTrail captura todas as chamadas de API para o IAM Access Analyzer como eventos. As chamadas capturadas incluem chamadas do console do IAM Access Analyzer e chamadas de código para as operações de API do IAM Access Analyzer.

Se você criar uma trilha, poderá habilitar a entrega contínua de eventos do CloudTrail para um bucket do Amazon S3, incluindo eventos para o IAM Access Analyzer. Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes no console do CloudTrail no Histórico de eventos.

Usando as informações coletadas pelo CloudTrail, é possível determinar a solicitação feita ao IAM Access Analyzer, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita, além de detalhes adicionais.

Para saber mais sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.

Informações do IAM Access Analyzer no CloudTrail

O CloudTrail é habilitado em sua conta AWS ao criá-la. Quando a atividade ocorre no IAM Access Analyzer, essa atividade é registrada em um evento do CloudTrail junto com outros eventos de produtos da AWS no Event history (Histórico de eventos). É possível visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com o histórico de eventos do CloudTrail.

Para obter um registro contínuo de eventos em sua conta da AWS, incluindo eventos do IAM Access Analyzer, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. Por padrão, quando uma trilha é criada no console, a mesma é aplicada a todas as regiões da AWS. A trilha registra logs de eventos de todas as Regiões na AWS divisória e entrega os arquivos do log para o bucket Amazon S3 especificado. Além disso, é possível configurar outros AWS serviços para melhor analisar e agir de acordo com dados coletados do evento nos logs CloudTrail. Para saber mais, consulte:

- Visão Geral para Criar uma Trilha
- Serviços e integrações com suporte no CloudTrail
- Configurando Notificações Amazon SNS para CloudTrail
- Recebendo Arquivos de Log CloudTrail de Várias Regiões e Recebendo Arquivos de Log CloudTrail de Várias Contas

Todas as ações do IAM Access Analyzer são registradas pelo CloudTrail e documentadas em IAM Access Analyzer API Reference (Referência da API do IAM Access Analyzer). Por exemplo, as chamadas às ações `CreateAnalyzer`, `CreateArchiveRule` e `ListFindings` geram entradas nos arquivos de log do CloudTrail.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário-raiz ou usuário do AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro serviço da AWS.

Para saber mais, consulte o Elemento userIdentity do CloudTrail.

Noções básicas sobre registros de arquivo de log do IAM Access Analyzer

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket Amazon S3 especificado. Os arquivos de log do CloudTrail contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte, e inclui informações sobre a ação solicitada, data e hora da ação, parâmetros da solicitação e assim por diante. Os arquivos de log do CloudTrail não são um rastreamento de pilha ordenada de chamadas de API pública, portanto, não são exibidos em uma ordem específica.

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra a operação `CreateAnalyzer` realizada por uma sessão de função assumida de nome `Alice-tempcreds` em “14 de junho de 2021”. A sessão de função foi emitida pela função de nome `admin-tempcreds`.

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
```

```
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Chaves de filtro do IAM Access Analyzer

É possível usar as chaves de filtro abaixo para definir uma regra de arquivamento (`CreateArchiveRule`), atualizar uma regra de arquivamento (`UpdateArchiveRule`), recuperar uma lista de descobertas (`ListFindings` e `ListFindingsV2`) ou recuperar uma lista de descobertas de pré-visualização de acesso para um recurso (`ListAccessPreviewFindings`).

Não há diferença entre usar a API do IAM e o CloudFormation para configurar regras de arquivamento.

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
do recurso	Recurso	O ARN identifica exclusivamente o recurso ao qual o principal externo tem acesso. Para saber mais, consulte Nomes de recursos da Amazon (ARNs).	String	Sim	Yes (Sim	Yes (Sim	Externo Interno Não utilizado
resourceType `AWS::S3 Bucket` | `AWS::IAM :Role` | `AWS::SQS :Queue` | `AWS::Lambda::Function` |	Tipo de recurso	O tipo de recurso ao qual o principal externo tem acesso. **Note** Os analisadores de acesso interno não são compatíveis com	String	Sim	Yes (Sim	Yes (Sim	Externo Interno Não utilizado

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
`AWS::Lambda::LayerVersion` |`AWS::KMS::Key` | `AWS::SecretsManager::Secret` | `AWS::EFS::FileSystem` | `AWS::EC2::Snapshot` | `AWS::ECR::Repository` | `AWS::RDS::DBSnapshot` | `AWS::RDS::DBClusterSnapshot` | `AWS::SNS::Topic` |		todos os tipos de recursos que são compatíveis com os analisadores de acesso externo. Os analisadores de acessos não utilizados oferecem suporte apenas a perfis e usuários do IAM. Para obter mais informações, consulte Tipos de recursos compatíve					

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
`AWS::S3Directory|cket | AWS::DynamoDB::Table | AWS::DynamoDB::Stream | AWS::IAM:User`		is com o IAM Access Analyzer para acesso externo e interno.					
resourceOwnerAccount	Conta proprietária do recurso	O ID de 12 dígitos da conta da AWS que possui o recurso. Para saber mais, consulte Identificadores de conta da AWS.	String	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	Externo Interno Não utilizado
isPublic	Acesso público	Indica se a descoberta relata um recurso que tem uma política que permite o acesso público.	Booleano	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
findingType `ExternalAccess` | `UnusedIAMRole` | `UnusedIAMUserAccessKey` | `UnusedIAMUserPassword` | `UnusedPermission` | `InternalAccess`	Tipo de descoberta	O tipo da descoberta. Para analisadores de acesso externo, o tipo é `ExternalAccess`. Para analisadores de acesso não utilizado, o tipo pode ser `UnusedIAMRole`, `UnusedIAMUserAccessKey`, `UnusedIAMUserPassword` ou `UnusedPermission`. Para analisadores de acesso interno, o tipo é `InternalAccess`.	String	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	Externo Interno Não utilizado

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
resourceControlPolicyRestriction `APPLIED | APPLICABLE | FAILED_ _EVALUAT _RCP | NOT_APPLICABLE`	Restrição da política de controle de recursos (RCP	O tipo de restrição aplicada pelo proprietário do recurso com uma política de controle de recursos (RCP) do Organizations. Para obter mais informações sobre os valores dessa chave de filtro, consulte ExternalAccessDetails e InternalAccessDetails na IAM Access Analyzer API Reference.	String	Sim	Yes (Sim	Yes (Sim	Externo Interno

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
serviceControlPolicyRestriction `APPLIED | APPLICABLE | FAILED_EVALUATING_SCP | NOT_APPLICABLE`	Restrição da política de controle de serviços (SCP)	O tipo de restrição aplicada por uma política de controle de serviços (SCP) do Organizations. Para obter mais informações sobre os valores dessa chave de filtro, consulte InternalAccessDetails na IAM Access Analyzer API Reference.	String	Sim	Yes (Sim	Yes (Sim	Interno
status `ACTIVE | ARCHIVED | RESOLVED`	Status	O status atual da descoberta.	String	No (Não	Yes (Sim	Yes (Sim	Externo Interno Não utilizado
Erro	Erro	Indica o erro relatado para a descoberta.	String	Sim	Yes (Sim	Yes (Sim	Externo Interno

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
principal.AWS	AWS Conta da	A conta que recebeu acesso ao recurso no campo `Principal` da descoberta. Insira o ID de 12 dígitos da conta AWS ou o ARN do usuário ou função externa da AWS. Para saber mais, consulte Identificadores de conta da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo
principal.Federated	Usuário federado	O ARN da identidade federada que tem acesso ao recurso na descoberta. Para saber mais, consulte Federação e provedores de identidade	String	Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition.aws:PrincipalArn	ARN da entidade principal	O ARN da entidade principal (usuário, perfil ou grupo do IAM) indicado como a condição para acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo
condition.aws:PrincipalOrgID	OrgID da entidade principal	O identificador da organização do principal indicado como a condição para o acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition .aws:PrincipalOrgPaths	OrgPaths da entidade principal	O ID da organização ou da unidade organizacional (UO) indicada como a condição para acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo
condition .aws:SourceIp	IP de origem	O endereço IP que permite ao principal acesso ao recurso ao usar o endereço IP especificado. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	Endereço IP	Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition .aws:SourceVpc	VPC de origem	O ID da VPC que permite ao principal acesso ao recurso ao usar a VPC especificada. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo
condition .aws:UserId	ID de usuário	O ID do usuário do IAM de uma conta externa indicada como a condição de acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition .aws:VpceAccount	Conta da VPCE	O ID da conta do endpoint da VPC que permite a entidade principal acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo Interno
condition .aws:SourceVpcArn	ARN da VPC de origem	O ARN da VPC que permite à entidade principal acesso ao recurso ao usar a VPC especificada. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	ARN	Yes (Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition .aws:VpceOrgID	OrgID da VPCE	O ID organizacional do endpoint da VPC que permite a entidade principal acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String	Sim	Yes (Sim	Yes (Sim	Externo Interno
condition .aws:VpceOrgPaths	OrgPaths da VPCE	A unidade organizacional (UO) do endpoint da VPC que permite a entidade principal acesso ao recurso. Para saber mais, consulte Chaves de contexto de condição globais da AWS.	String (lista)	Yes (Sim	Yes (Sim	Yes (Sim	Externo Interno

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition .cognito-identity. amazonav .com:aud	Público do Cognito	O ID do grupo de identidades do Amazon Cognito especificado como uma condição para o acesso à função do IAM na descoberta. Para saber mais, consulte Chaves de contexto de condição do IAM e do AWS STS.	String	Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition.graph.facebook.com:app_id	ID da aplicação do Facebook	O ID da aplicação do Facebook (ou o ID do site) especificado como uma condição para permitir o acesso à federação do Login with Facebook à função do IAM na descoberta. Para saber mais, consulte Chaves de contexto de condição do IAM e do AWS STS.	String	Sim	Yes (Sim	Yes (Sim	Externo
condition.accounts.google.com:aud	Público do Google	O ID da aplicação do Google especificado como uma condição para o acesso à função do IAM. Para saber mais, consulte Chaves de contexto de condição do IAM e do AWS STS.	String	Sim	Yes (Sim	Yes (Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
condition .kms:CallerAccount	ID da chave do KMS	O ID da conta da AWS que possui a entidade que faz a chamada (usuário, perfil ou usuário-raiz da conta do IAM) usada por serviços que chamam o AWS KMS. Para saber mais, consulte Chaves de condição para AWS Key Management Service.	String	Sim	Yes (Sim	Yes (Sim	Externo
condition .www.amazon.com:app_id	ID da aplicação da Amazon	O ID da aplicação da Amazon (ou o ID do site) especificada como uma condição para permitir o acesso à federação do Login with Amazon à função. Para saber mais, consulte	String	Sim	Yes (Sim	Yes (Sim)	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
ID de	ID* da descoberta	O ID da descoberta.	String	⊗ No (Não	⊙ Yes (Sim	⊙ Yes (Sim	Externo Interno Não utilizado
changeType `CHANGED | NEW | UNCHANGI`		Fornece contexto sobre como a descoberta de pré-visualização de acesso se compara ao acesso existente identificado no IAM Access Analyzer.	String	⊗ No (Não	⊗ No (Não	⊙ Sim	Externo
existingFindingId		O ID existente da descoberta no IAM Access Analyzer, fornecido apenas para descobertas existentes na pré-visualização de acesso.	String	⊗ No (Não	⊗ No (Não	⊙ Sim	Externo

Criterion	Campo Console de gerenciamento da AWS	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso	Tipos de analisadores compatíveis
existingFindingStatus		O status existente da descoberta, fornecido apenas para descobertas existentes na pré-visualização de acesso.	String	⊗ No (Não	⊗ No (Não	⊙ Sim	Externo

Usar perfis vinculados a serviço do AWS Identity and Access Management Access Analyzer

O AWS Identity and Access Management Access Analyzer usa um perfil vinculado ao serviço do IAM. Uma função vinculada a serviço é um tipo exclusivo de perfil do IAM vinculada diretamente ao IAM Access Analyzer. As funções vinculadas a um serviço são predefinidas pelo IAM Access Analyzer e incluem todas as permissões que o atributo exige para chamar outros serviços AWS em seu nome.

Uma função vinculada a um serviço facilita a configuração do IAM Access Analyzer porque não é preciso adicionar as permissões necessárias manualmente. O IAM Access Analyzer define as permissões de suas funções vinculadas a um serviço e, exceto se definido de outra forma, somente o IAM Access Analyzer pode assumir suas funções. As permissões definidas incluem a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

Para obter informações sobre outros serviços compatíveis com perfis vinculados a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que contenham Sim na coluna Perfil vinculado a serviço. Escolha Sim com um link para visualizar a documentação da função vinculada a esse serviço.

Permissões de perfil vinculado ao serviço do AWS Identity and Access Management Access Analyzer

O AWS Identity and Access Management Access Analyzer usa a função vinculada a serviço chamada AWSServiceRoleForAccessAnalyzer, que permite que o Access Analyzer analise os metadados do recurso para acessos externos e para analisar a atividade e identificar acessos não utilizados.

A função vinculada ao serviço AWSServiceRoleForAccessAnalyzer confia nos seguintes serviços para assumir a função:

- `access-analyzer.amazonaws.com`

A política de permissões da função chamada `AccessAnalyzerServiceRolePolicy` permite que o IAM Access Analyzer conclua ações em recursos específicos.

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua um perfil vinculado a serviço. Para saber mais, consulte Permissões de função vinculada a serviço no Guia do usuário do IAM.

Criar uma função vinculada a um serviço para o IAM Access Analyzer

Não é necessário criar manualmente um perfil vinculado ao serviço. Quando você habilita o Access Analyzer no Console de gerenciamento da AWS ou na API da AWS, o IAM Access Analyzer cria a função vinculada ao serviço para você. A mesma função vinculada ao serviço é usada em todas as regiões nas quais você habilita o IAM Access Analyzer. Tanto as descobertas de acessos externos quanto as de acessos não utilizados usam a mesma função vinculada ao serviço.

> **Note**
>
> O IAM Access Analyzer é regional. É necessário habilitar o IAM Access Analyzer em cada região de maneira independente.

Se excluir essa função vinculada ao serviço, o IAM Access Analyzer recriará a função na próxima vez em que você criar um analisador.

Você também pode usar o console do IAM para criar uma função vinculada ao serviço com o caso de uso do Access Analyzer. Na AWS CLI ou na API do AWS, crie um perfil vinculado a serviço com

o nome de serviço `access-analyzer.amazonaws.com`. Para obter mais informações, consulte Criar uma função vinculada ao serviço no Guia do usuário do IAM. Se você excluir essa função vinculada ao serviço, será possível usar esse mesmo processo para criar a função novamente.

Editar uma função vinculada a um serviço para o IAM Access Analyzer

O IAM Access Analyzer não permite editar a função vinculada ao serviço AWSServiceRoleForAccessAnalyzer. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para saber mais, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Excluir uma função vinculada a um serviço para o IAM Access Analyzer

Se você não precisar mais usar um atributo ou serviço que requer um perfil vinculado ao serviço, é recomendável excluí-la. Dessa forma, você não terá uma entidade não utilizada que não seja monitorada ativamente ou mantida. No entanto, você deve limpar os atributos de seu perfil vinculado ao serviço antes de excluí-lo manualmente.

Se o IAM Access Analyzer estiver habilitado em uma ou mais regiões da sua AWS Organizations, você deverá excluir todos os analisadores de todas as regiões da organização antes de tentar excluir esse perfil.

> **Note**
>
> Se o IAM Access Analyzer estiver usando a função quando você tentar excluir os recursos, a exclusão poderá falhar. Se isso acontecer, espere alguns minutos e tente a operação novamente.

Para excluir os recursos do IAM Access Analyzer usados pelo perfil AWSServiceRoleForAccessAnalyzer

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Na seção Access reports (Relatórios de acesso), em Access analyzer (Analisador de acesso), selecione Analyzers (Analisadores).
3. Escolha o analisador do qual deseja excluir os recursos do IAM Access Analyzer anexados ao perfil vinculado ao serviço.
4. Escolha Excluir.

5. Para confirmar que deseja excluir os analisadores, insira **`delete`** e selecione Delete (Excluir).

Como excluir manualmente o perfil vinculado ao serviço usando o IAM

Use o console do IAM, a AWS CLI ou a API da AWS para excluir a função vinculada a serviço AWSServiceRoleForAccessAnalyzer. Para saber mais, consulte Excluir um perfil vinculado ao serviço no Guia do usuário do IAM.

Regiões compatíveis com funções vinculadas a serviço do IAM Access Analyzer

O IAM Access Analyzer oferece suporte a funções vinculadas a serviços em todas as regiões onde o serviço está disponível. Para obter mais informações, consulte Regiões e endpoints da AWS.

Pré-visualizar o acesso

Além de ajudar você a identificar recursos que são compartilhados com uma entidade externa, o AWS IAM Access Analyzer também mostra uma pré-visualização das descobertas do IAM Access Analyzer antes de implantar permissões de recursos para que você possa confirmar se suas alterações na política concedem ao seu recurso somente o acesso público e entre contas desejado. Isso ajuda você a começar a usar o acesso externo pretendido aos seus recursos.

Você pode pré-visualizar e validar o acesso público e entre contas aos buckets do Amazon S3 no console do Amazon S3. Você também pode usar as APIs do IAM Access Analyzer para pré-visualizar o acesso público e entre contas de seus buckets do Amazon S3, chaves do AWS KMS, perfis do IAM, filas do Amazon SQS e segredos do Secrets Manager fornecendo permissões propostas para seu recurso.

Tópicos

- Pré-visualização de acesso no console do Amazon S3
- Pré-visualização de acesso com APIs do IAM Access Analyzer

Pré-visualização de acesso no console do Amazon S3

Depois de concluir sua política de bucket no console do Amazon S3, você terá a opção de pré-visualizar o acesso público e entre contas ao bucket do Amazon S3. Você pode validar se suas alterações de política concedem apenas acesso externo pretendido antes de escolher a opção Save changes (Salvar alterações). Essa etapa opcional permite pré-visualizar descobertas do AWS Identity and Access Management Access Analyzer para seu bucket. Você pode validar se a alteração de

política introduz novas descobertas ou resolve descobertas existentes para acesso externo. Você pode ignorar essa etapa de validação e salvar sua política de bucket do Amazon S3 a qualquer momento.

Para pré-visualizar o acesso externo ao seu bucket, você deve ter um analisador de conta ativo na região do bucket com a conta como a zona de confiança. Você também deve ter as permissões necessárias para usar o IAM Access Analyzer e pré-visualizar o acesso. Para obter mais informações sobre como habilitar o IAM Access Analyzer e as permissões necessárias, consulte Conceitos básicos do AWS Identity and Access Management Access Analyzer.

Para pré-visualizar o acesso ao bucket do Amazon S3 ao criar ou editar sua política de bucket

1. Quando terminar de criar ou editar sua política de bucket, certifique-se de que a política seja uma política de bucket do Amazon S3 válida. O ARN da política deve corresponder ao ARN do bucket, e os elementos da política devem ser válidos.
2. Abaixo da política, em Preview external access (Pré-visualizar acesso externo), escolha um analisador de conta ativo e, em seguida, escolha Preview (Pré-visualizar). Uma pré-visualização das descobertas do IAM Access Analyzer é gerada para o seu bucket. A pré-visualização analisa a política de bucket do Amazon S3 exibida, juntamente com as permissões de bucket existentes. Isso inclui as configurações de BPA de bucket e conta, ACL de bucket, pontos de acesso do Amazon S3 e pontos de acesso multirregiões anexados ao bucket e suas políticas e configurações de BPA.
3. Quando a pré-visualização de acesso for concluída, uma pré-visualização das descobertas do IAM Access Analyzer será exibida. Cada descoberta relata uma instância de uma entidade fora da conta, com acesso ao seu bucket depois de salvar a política. Você pode validar o acesso ao seu bucket revisando cada descoberta. O cabeçalho da descoberta fornece um resumo do acesso, e você pode expandir a descoberta para revisar seus respectivos detalhes. Os distintivos de descoberta fornecem contexto sobre como salvar a política de bucket alteraria o acesso ao bucket. Por exemplo, eles ajudam a confirmar se a alteração de política introduz novas descobertas ou se resolve descobertas existentes para acesso externo:

    a. New (Novo): indica uma descoberta de novo acesso externo que a política introduzirá.

    b. Resolved (Resolvido): indica uma descoberta de acesso externo existente que a política removerá.

    c. Archived (Arquivado): indica uma descoberta de novo acesso externo que seria arquivado automaticamente, com base nas regras de arquivamento do analisador que definem quando as descobertas devem ser marcadas como pretendidas.

d. Existing (Existente): indica uma descoberta existente de acesso externo que permanecerá inalterada.

e. Public (Público): se uma descoberta for de acesso público ao recurso, ela terá um distintivo Public (Público), além de um dos distintivos acima.

4. Se você identificar o acesso externo que não pretende introduzir ou remover, poderá revisar a política e escolher Preview (Pré-visualizar) novamente até obter o acesso externo desejado. Se você tiver uma descoberta rotulada como Public (Pública), recomendamos revisar a política para remover o acesso público antes de escolher a opção Save changes (Salvar alterações). A pré-visualização do acesso é uma etapa opcional, e você pode escolher a opção Save changes (Salvar alterações) a qualquer momento.

Pré-visualização de acesso com APIs do IAM Access Analyzer

Você pode usar APIs do IAM Access Analyzer para pré-visualizar o acesso público e entre contas para seus buckets do Amazon S3, chaves do AWS KMS, perfis do IAM, filas do Amazon SQS e segredos do Secrets Manager. Você pode pré-visualizar o acesso fornecendo permissões propostas para um recurso existente que você possui ou um novo recurso que deseja implantar.

Para pré-visualizar o acesso externo ao seu recurso, você deve ter um analisador de conta ativo para a conta e a região do recurso. Você também deve ter as permissões necessárias para usar o IAM Access Analyzer e pré-visualizar o acesso. Para obter mais informações sobre como habilitar o IAM Access Analyzer e as permissões necessárias, consulte Conceitos básicos do AWS Identity and Access Management Access Analyzer.

Para pré-visualizar o acesso de um recurso, você pode usar a operação `CreateAccessPreview` e fornecer o ARN do analisador e a configuração de controle de acesso do recurso. O serviço retorna o ID exclusivo para a pré-visualização de acesso, que você pode usar para conferir o status da pré-visualização de acesso com a operação `GetAccessPreview`. Quando o status for `Completed`, você poderá usar a operação `ListAccessPreviewFindings` para recuperar as descobertas geradas para a pré-visualização de acesso. As operações `GetAccessPreview` e `ListAccessPreviewFindings` recuperarão pré-visualizações de acesso e descobertas criadas em cerca de 24 horas.

Cada descoberta recuperada contém detalhes da descoberta descrevendo o acesso. Um status de pré-visualização da descoberta que descreve se a descoberta seria `Active`, `Archived` ou `Resolved` após a implantação de permissões e um `changeType`. O `changeType` fornece contexto

comparativo entre a descoberta da pré-visualização de acesso e o acesso existente identificado no IAM Access Analyzer:

- New (Novo): a descoberta é de um acesso recém-introduzido.
- Unchanged (Inalterado): a descoberta de pré-visualização é uma descoberta existente que permanecerá inalterada.
- Changed (Alterado): a descoberta de pré-visualização é uma descoberta existente com uma alteração no status.

O `status` e o `changeType` ajudam a entender como a configuração do recurso mudará o acesso ao recurso existente. Se o `changeType` for `Unchanged` (Inalterado) ou Changed (Alterado), a descoberta também conterá o ID existente e o status da descoberta no IAM Access Analyzer. Por exemplo, uma descoberta `Changed` com status de pré-visualização `Resolved` e status `Active` existente indica que a descoberta `Active` existente do recurso se tornaria `Resolved` como resultado da alteração de permissões propostas.

Você pode usar a operação `ListAccessPreviews` para recuperar uma lista de pré-visualizações de acesso para o analisador especificado. Essa operação recuperará informações sobre a pré-visualização de acesso criada em cerca de uma hora.

Em geral, se a pré-visualização de acesso for para um recurso existente e você deixar uma opção de configuração não especificada, a pré-visualização de acesso usará a configuração de recurso existente por padrão. Se a pré-visualização de acesso for para um novo recurso e você deixar uma opção de configuração não especificada, a pré-visualização de acesso usará o valor padrão dependendo do tipo de recurso. Para casos de configuração para cada tipo de recurso, consulte abaixo.

Pré-visualizar o acesso ao bucket do Amazon S3

Para criar uma pré-visualização de acesso para um novo bucket do Amazon S3 ou um bucket existente do Amazon S3 que você possui, você pode propor uma configuração de bucket especificando a política de bucket do Amazon S3, ACLs de bucket, configurações de BPA de bucket e pontos de acesso do Amazon S3, incluindo pontos de acesso multirregiões, anexados ao bucket.

> **Note**
>
> Antes de tentar criar uma pré-visualização de acesso para um novo bucket, recomendamos que você chame a operação HeadBucket do Amazon S3 para conferir se o bucket nomeado

já existe. Essa operação é útil para determinar se existe um bucket e se você tem permissão para acessá-lo.

Bucket policy (Política de bucket): se a configuração for para um bucket existente do Amazon S3 e você não especificar a política de bucket do Amazon S3, a pré-visualização de acesso usará a política existente anexada ao bucket. Se a visualização de acesso for para um novo recurso e você não especificar a política de bucket do Amazon S3, a pré-visualização de acesso assumirá um bucket sem uma política. Para propor a exclusão de uma política de bucket existente, você pode especificar uma string vazia. Para obter mais informações sobre limites de política de bucket com suporte, consulte Exemplos de políticas de bucket.

Bucket ACL grants (Concessões da ACL do bucket): você pode propor até 100 concessões de ACL por bucket. Se a configuração de concessão proposta for para um bucket existente, a pré-visualização de acesso usará a lista proposta de configurações de concessão no lugar das concessões existentes. Caso contrário, a pré-visualização de acesso usará as concessões existentes para o bucket.

Bucket access points (Pontos de acesso do bucket): a análise dá suporte para até 100 pontos de acesso, incluindo pontos de acesso multirregiões, por bucket, incluindo até dez novos pontos de acesso que você pode propor por bucket. Se a configuração de ponto de acesso proposta do Amazon S3 for para um bucket existente, a pré-visualização de acesso usará a configuração de ponto de acesso proposta no lugar dos pontos de acesso existentes. Para propor um ponto de acesso sem uma política, você pode fornecer uma string vazia como a política de ponto de acesso. Para obter mais informações sobre limites de política de ponto de acesso, consulte Restrições e limitações de pontos de acesso.

Block public access configuration (Configuração de bloqueio de acesso público): se a configuração proposta for para um bucket existente do Amazon S3 e você não especificar a configuração, a pré-visualização de acesso usará a configuração existente. Se a configuração proposta for para um novo bucket e você não especificar a configuração de BPA do bucket, a pré-visualização de acesso usará `false`. Se a configuração proposta for para um novo ponto de acesso ou ponto de acesso multirregiões e você não especificar a configuração de BPA do ponto de acesso, a pré-visualização de acesso usará `true`.

Pré-visualizar o acesso à chave do AWS KMS

Para criar uma pré-visualização de acesso para uma nova chave do AWS KMS ou uma chave existente do AWS KMS que você possui, você pode propor uma configuração de chave do AWS KMS especificando a política de chave e a configuração de concessão do AWS KMS.

Política de chave do AWS KMS: se a configuração for para uma chave existente e a política de chave não for especificada, a pré-visualização de acesso usará a política existente para a chave. Se a pré-visualização de acesso for para um novo recurso e você não especificar a política de chave, a pré-visualização de acesso usará a política de chave padrão. A política de chave proposta não pode ser uma string vazia.

AWS KMS grants (Concessões do KMS): a análise dá suporte para até 100 concessões do KMS por configuração*.* Se a configuração de concessão proposta for para uma chave existente, a pré-visualização de acesso usará a lista proposta de configurações de concessão no lugar das concessões existentes. Caso contrário, a pré-visualização de acesso usará as concessões existentes para a chave.

Pré-visualizar o acesso à função do IAM

Para criar uma pré-visualização de acesso para uma nova função do IAM ou uma função existente do IAM de sua propriedade, você pode propor uma configuração de função do IAM especificando a política de confiança.

Role trust policy (Política de confiança da função): se a configuração for para uma nova função do IAM, será necessário especificar a política de confiança. Se a configuração for para uma função do IAM existente que você possui e você não propuser a política de confiança, a pré-visualização de acesso usará a política de confiança existente para a função. A política de confiança proposta não pode ser uma string vazia.

Pré-visualizar o acesso à sua fila do Amazon SQS

Para criar uma pré-visualização de acesso para uma nova fila do Amazon SQS ou uma fila existente do Amazon SQS que você possui, você pode propor uma configuração de fila do Amazon SQS especificando a política do Amazon SQS para a fila.

Amazon SQS queue policy (Política de fila do Amazon SQS): se a configuração for para uma fila existente do Amazon SQS e você não especificar a política do Amazon SQS, a pré-visualização de acesso usará a política existente do Amazon SQS para a fila. Se a pré-visualização de acesso for

para um novo recurso e você não especificar a política, a pré-visualização de acesso assumirá uma fila do Amazon SQS sem uma política. Para propor a exclusão de uma política de fila existente do Amazon SQS, você pode especificar uma string vazia para a política do Amazon SQS.

Pré-visualizar o acesso ao segredo do Secrets Manager

Para criar uma pré-visualização de acesso para um novo segredo do Secrets Manager ou um segredo existente do Secrets Manager que você possui, você pode propor uma configuração de segredo do Secrets Manager especificando a política de segredo e uma chave de criptografia opcional do AWS KMS.

Secret policy (Política de segredo): se a configuração for para um segredo existente e você não especificar a política de segredo, a pré-visualização de acesso usará a política existente para o segredo. Se a pré-visualização de acesso for para um novo recurso e você não especificar a política, a pré-visualização de acesso assumirá um segredo sem uma política. Para propor a exclusão de uma política existente, você pode especificar uma string vazia.

Chave de criptografia do AWS KMS: se a configuração proposta for para um novo segredo e a ID da chave do AWS KMS não for especificada, a pré-visualização de acesso usará a chave do KMS padrão da conta da AWS. Se você especificar uma string vazia para o ID da chave do AWS KMS, a pré-visualização de acesso usará a chave do KMS padrão da conta do AWS.

Verificações de validação de políticas

O IAM Access Analyzer fornece verificações de políticas que ajudam a validar suas políticas do IAM antes de anexá-las a uma entidade. Isso inclui verificações básicas de políticas fornecidas pela validação de políticas para validar sua política em relação à gramática da política e às práticas recomendadas da AWS. É possível visualizar as descobertas de verificação de validação de política que incluem avisos de segurança, erros, avisos gerais e sugestões para sua política.

Você pode usar verificações de políticas personalizadas para verificar novos acessos com base em seus padrões de segurança. Uma cobrança é associada a cada verificação de novo acesso. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Como funcionam as verificações de políticas personalizadas

Você pode validar suas políticas de acordo com seus padrões especificados de segurança usando verificações de política personalizadas do AWS Identity and Access Management Access Analyzer. É possível executar os seguintes tipos de verificações de política personalizada:

- Compare uma política de referência: ao editar uma política, você pode verificar se a política atualizada concede novo acesso em comparação com uma política de referência, como uma versão existente da política. É possível executar essa verificação ao editar uma política usando a AWS Command Line Interface (AWS CLI), a API do IAM Access Analyzer (API), ou um editor de políticas JSON no console IAM.

  > **Note**
  >
  > As verificações de políticas personalizadas do IAM Access Analyzer permitem curingas no elemento `Principal` para políticas de recursos de referência.

- Comparar com uma lista de ações ou recursos do IAM: você pode verificar para garantir que ações ou recursos específicos do IAM não são permitidos por sua política. Se especificar somente ações, o IAM Access Analyzer verificará o acesso às ações em todos os recursos da política. Se especificar somente recursos, o IAM Access Analyzer verificará quais ações têm acesso aos recursos especificados. Se especificar ações e recursos, o IAM Access Analyzer verificará quais das ações especificadas têm acesso aos recursos especificados. É possível executar essa verificação ao criar ou editar uma política usando a AWS CLI ou a API.
- Verificar o acesso público: você pode verificar se uma política de recursos pode conceder acesso público a um tipo de recurso específico. É possível executar essa verificação ao criar ou editar uma política usando a AWS CLI ou a API. Esse tipo de verificação de política personalizada é diferente da visualização prévia do acesso, porque a verificação não exige nenhum contexto de analisador de conta ou de acesso externo. As visualizações prévias de acesso permitem que você visualize as descobertas do IAM Access Analyzer antes de implantar permissões de recursos, enquanto a verificação personalizada determina se uma política pode conceder acesso público.

Uma cobrança é associada a cada verificação de política personalizada. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Você pode executar verificações de políticas personalizadas em políticas baseadas em identidade e recursos. As verificações de políticas personalizadas não dependem de técnicas de correspondência de padrões nem da análise de logs de acesso para determinar se um acesso novo ou específico é permitido por uma política. Semelhante às descobertas de acessos externos, as verificações de políticas personalizadas são baseadas em Zelkova. O Zelkova converte políticas do IAM em declarações lógicas equivalentes e executa um conjunto de solucionadores lógicos especializados e de uso geral (teorias do módulo da satisfatibilidade) em relação ao problema. Para verificar o acesso novo ou especificado, o IAM Access Analyzer aplica Zelkova repetidamente a uma política.

As consultas se tornam cada vez mais específicas para caracterizar classes de comportamentos que a política permite com base no conteúdo da política. Para obter mais informações sobre as teorias do módulo da satisfatibilidade, consulte Satisfiability Modulo Theories.

Em casos raros, o IAM Access Analyzer não é capaz de determinar completamente se uma instrução de política concede acesso novo ou especificado. Nesses casos, ele erra ao declarar um falso positivo ao falhar na verificação da política personalizada. O IAM Access Analyzer foi projetado para fornecer uma avaliação de política completa e faz o possível para minimizar a ocorrência de falsos negativos. Essa abordagem significa que o IAM Access Analyzer fornece um alto grau de garantia de que uma verificação aprovada significa que o acesso não foi concedido pela política. Você pode inspecionar manualmente as verificações que falharam ao revisar a declaração de política relatada na resposta do IAM Access Analyzer.

Exemplos de políticas de referência para verificar novos acessos

Você pode encontrar exemplos de políticas de referência e aprender como configurar e executar uma verificação de política personalizada para novos acessos no repositório de amostras de verificações de políticas personalizadas do IAM Access Analyzer no GitHub.

Antes de usar esses exemplos

Antes de usar esses exemplos de políticas de referência, faça o seguinte:

- Revise atentamente e personalize as políticas de referência de acordo com suas necessidades específicas.
- Teste detalhadamente as políticas de referência em seu ambiente com os Serviços da AWS que você usa.

  As políticas de referência demonstram a implementação e o uso de verificações de políticas personalizadas. Eles não são destinados a serem interpretados como recomendações oficiais ou práticas recomendadas da AWS a serem implementadas exatamente como mostrado. É sua responsabilidade testar cuidadosamente as políticas de referência quanto à sua adequação para resolver os requisitos de segurança do seu ambiente.
- As verificações de políticas personalizadas são independentes do ambiente em suas análises. Sua análise considera apenas as informações contidas nas políticas de entrada. Por exemplo, verificações de políticas personalizadas não podem verificar se uma conta é membro de uma organização específica AWS. Portanto, as verificações

de políticas personalizadas não podem comparar novos acessos com base nos valores da chave de condição para as chaves de condição `aws:PrincipalOrgId` e `aws:PrincipalAccount`.

Inspecionar verificações de políticas personalizadas com falha

Quando uma verificação de política personalizada falha, a resposta do IAM Access Analyzer inclui o ID da declaração (`Sid`) da declaração de política que causou a falha na verificação. Embora a ID da declaração seja um elemento opcional da política, recomendamos que você adicione uma ID da declaração para cada declaração de política. A verificação personalizada de política também retorna um índice de declaração para ajudar a identificar o motivo da falha na verificação. O índice da declaração segue a numeração com base em zero, em que a primeira declaração é referenciada como 0. Quando há várias declarações que causam a falha de uma verificação, a verificação retorna somente uma ID de declaração por vez. Recomendamos que você corrija a declaração destacada no motivo e execute novamente a verificação até que ela seja aprovada.

Validar políticas com o IAM Access Analyzer

É possível validar suas políticas usando validações de política do AWS Identity and Access Management Access Analyzer. É possível criar ou editar uma política usando a AWS CLI,a API da AWS ou o editor de políticas de JSON no console do IAM. O IAM Access Analyzer valida sua política em relação à gramática da política do IAM e às práticas recomendadas da AWS. É possível visualizar as descobertas de verificação de validação de política que incluem avisos de segurança, erros, avisos gerais e sugestões para sua política. Essas descobertas fornecem recomendações práticas que ajudam a criar políticas que sejam funcionais e estejam em conformidade com as práticas recomendadas de segurança. Para visualizar uma lista das verificações de política básicas executadas pelo IAM Access Analyzer, consulte Referência para verificação de validação de política do IAM.

Validar políticas no IAM (console)

Você pode visualizar as descobertas geradas pela validação de política do IAM Access Analyzer ao criar ou editar uma política gerenciada no console do IAM. Você também pode visualizar essas descobertas para políticas de usuário ou função em linha. O IAM Access Analyzer não gera essas descobertas para políticas de grupo em linha.

Para visualizar descobertas geradas por verificações de política para políticas de JSON do IAM

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Comece criando ou editando uma política usando um dos seguintes métodos:

   a. Para criar uma nova política gerenciada, acesse a página Policies (Políticas) e crie uma nova política. Para obter mais informações, consulte Criar políticas usando o editor de JSON.

   b. Para visualizar as verificações de política para uma política gerenciada pelo cliente existente, acesse a página Políticas, escolha o nome de uma política e, em seguida, selecione Editar. Para obter mais informações, consulte Edição de políticas gerenciadas pelo cliente (console).

   c. Para visualizar as verificações de política para uma política em linha em um usuário ou um perfil, acesse a página Usuários ou Perfis, escolha o nome de um usuário ou de um perfil, selecione o nome da política na guia Permissões e, em seguida, clique em Editar. Para obter mais informações, consulte Edição de políticas em linha (console).
3. No editor de política, escolha a guia JSON.
4. No painel de validação de política abaixo da política, escolha uma ou mais das guias a seguir. Os nomes das guias também indicam o número de cada tipo de descoberta para sua política.

   - Security (Segurança): exibe avisos se sua política permitir acesso que a AWS considera um risco de segurança porque o acesso é excessivamente permissivo.
   - Errors (Erros): exibe erros se a política incluir linhas que impeçam o funcionamento da política.
   - Avisos: exibe avisos se sua política não estiver em conformidade com as práticas recomendadas, mas os problemas não forem de riscos de segurança.
   - Suggestions (Sugestões): exibe sugestões se a AWS recomendar melhorias que não afetem as permissões da política.
5. Revise os detalhes da descoberta fornecidos pela verificação de política do IAM Access Analyzer. Cada descoberta indica a localização do problema relatado. Para saber mais sobre o que causa o problema e como resolvê-lo, escolha o link Saiba mais ao lado da descoberta. Você também pode pesquisar a verificação de política associada a cada descoberta na página de referência Verificações de políticas do Access Analyzer.
6. Opcional. Se você estiver editando uma política existente, poderá executar uma verificação de política personalizada para determinar se sua política atualizada concede novo acesso em

comparação com a versão existente. No painel de validação de política abaixo da política, escolha a guia Verificar novo acesso e, em seguida, escolha Verificar política. Se as permissões modificadas concederem novo acesso, a declaração será destacada no painel de validação da política. Se você não pretende conceder um novo acesso, atualize a declaração de política e escolha Verificar política até que nenhum novo acesso seja detectado. Para obter mais informações, consulte Validar políticas com verificações de políticas do IAM Access Analyzer.

> **Note**
>
> Uma cobrança é associada a cada verificação de novo acesso. Para obter mais detalhes sobre os preços, consulte Preços do IAM Access Analyzer.

7. Atualize sua política para resolver as descobertas.

> **Important**
>
> Teste políticas novas ou editadas cuidadosamente antes de implementá-las em seu fluxo de trabalho de produção.

8. Quando terminar, escolha Próximo. O Validado de políticas relata quaisquer erros de sintaxe que não sejam relatados pelo IAM Access Analyzer.

> **Note**
>
> É possível alternar entre as guias Visual e JSON sempre que quiser. Porém, se você fizer alterações ou escolher Avançar na guia Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

9. Para novas políticas, na página Revisar e criar, insira um Nome de política e uma Descrição (opcional) para a política que você está criando. Revise Permissões definidas nessa política para ver as permissões que são concedidas pela política. Em seguida, escolha Criar política para salvar seu trabalho.

   Para políticas existentes, na página Revisar e salvar, revise as Permissões definidas nessa política para ver as permissões concedidas pela sua política. Marque a caixa de seleção Definir esta nova versão como padrão. para salvar a versão atualizada como versão padrão da política. Em seguida escolha Salvar alterações para salvar o seu trabalho.

Validar políticas usando o IAM Access Analyzer (AWS CLI ou API da AWS)

Você pode visualizar as descobertas geradas pela validação de política do IAM Access Analyzer da AWS Command Line Interface (AWS CLI).

Para visualizar as descobertas geradas pela validação da política do IAM Access Analyzer (AWS CLI ou API da AWS).

Use uma das seguintes opções:

- AWS CLI: aws accessanalyzer validate-policy
- API da AWS:ValidatePolicy

Referência para verificação de validação de política do IAM

É possível validar suas políticas usando validações de política do AWS Identity and Access Management Access Analyzer. É possível criar ou editar uma política usando a AWS CLI, a API da AWS ou o editor de políticas de JSON no console do IAM. O IAM Access Analyzer valida sua política em relação à gramática da política do IAM e às práticas recomendadas da AWS. É possível visualizar as descobertas de verificação de validação de política que incluem avisos de segurança, erros, avisos gerais e sugestões para sua política. Essas descobertas fornecem recomendações práticas que ajudam a criar políticas que sejam funcionais e estejam em conformidade com as práticas recomendadas de segurança. A lista de verificações básicas de políticas fornecidas pelo IAM Access Analyzer é compartilhada abaixo. Não há cobrança adicional associada à execução das verificações de validação de política. Para obter mais informações sobre como validar políticas usando validação de política, consulte Validar políticas com o IAM Access Analyzer.

Erro: ARN account not allowed (Conta do ARN não permitida)

Código do problema: ARN_ACCOUNT_NOT_ALLOWED

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Resolver o erro

Remova o ID da conta do ARN do recurso. Os ARNs de recursos para alguns produtos da AWS não são compatíveis com a especificação de um ID de conta.

Por exemplo, o Amazon S3 não oferece suporte a um ID de conta como namespace em ARNs de bucket. Um nome de bucket do Amazon S3 é globalmente exclusivo, e o namespace é compartilhado por todas as contas da AWS. Para visualizar todos os tipos de recursos disponíveis no Amazon S3, consulte Tipos de recursos definidos pelo Amazon S3 naReferência de autorização do serviço.

Termos relacionados

- Recursos de políticas
- Identificadores de conta
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: ARN Region not allowed (Região do ARN não permitida)

Código do problema: ARN_REGION_NOT_ALLOWED

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Resolver o erro

Remova a região do ARN do recurso. Os ARNs de recursos para alguns produtos da AWS não oferecem suporte à especificação de uma região.

Por exemplo, o IAM é um serviço global. A parte da região de um ARN de recurso do IAM é sempre mantida em branco. Os recursos do IAM são globais, como uma conta da AWS é hoje. Por exemplo, depois de fazer login como usuário do IAM, você pode acessar produtos da AWS em qualquer região geográfica.

- Recursos de políticas
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: Data type mismatch (Incompatibilidade de tipo de dados)

Código do problema: DATA_TYPE_MISMATCH

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Resolver o erro

Atualize o texto para usar o tipo de dados com suporte.

Por exemplo, a chave de condição global `Version` requer um tipo de dados `String`. Se você fornecer uma data ou um número inteiro, o tipo de dados não corresponderá.

Termos relacionados

- Chaves de condições globais
- Elementos de política JSON do IAM: operadores de condição

Erro: Duplicate keys with different case (Teclas duplicadas com diferenciação de maiúsculas e minúsculas)

Código do problema: DUPLICATE_KEYS_WITH_DIFFERENT_CASE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Resolver o erro

Revise as chaves de condição semelhantes dentro do mesmo bloco de condição e use a mesma capitalização para todas as instâncias.

Um bloco de condição é o texto dentro do elemento `Condition`de uma instrução de política. Os nomes das chaves de condição não diferenciam maiúsculas de minúsculas. A diferenciação de maiúsculas e minúsculas dos valores da chave de condição depende do operador de condição utilizado. Para obter mais informações sobre diferenciação de maiúsculas e minúsculas em chaves de condição, consulte Elementos de política JSON do IAM: Condition.

Termos relacionados

- Condições

- Bloco de condição
- Chaves de condições globais
- AWS Chaves de condição do produto da

Erro: Invalid action (Ação inválida)

Código do problema: INVALID_ACTION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Resolver o erro

A ação especificada é inválida. Isso pode acontecer se você digitar incorretamente o prefixo do serviço ou o nome da ação. Para alguns problemas comuns, a verificação de política retorna uma ação sugerida.

Termos relacionados

- Ações de políticas
- AWS Ações do produto da

AWSPolíticas gerenciadas pela com esse erro

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

As seguintes políticas gerenciadas pela AWS incluem ações inválidas em suas instruções de política. Ações inválidas não afetam as permissões concedidas pela política. Ao usar uma política gerenciada pela AWS como referência para criar sua política gerenciada, a AWS recomenda que você remova as ações inválidas da sua política.

- AmazonEMRFullAccessPolicy_v2
- CloudWatchSyntheticsFullAccess

Erro: Invalid ARN account (Conta do ARN inválida)

Código do problema: INVALID_ARN_ACCOUNT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-
digit account ID."
```

Resolver o erro

Atualize o ID da conta no ARN do recurso. Os IDs de conta são números inteiros de 12 dígitos. Para saber como visualizar o ID da conta, consulte Localizar o ID da conta da AWS.

Termos relacionados

- Recursos de políticas
- Identificadores de conta
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: Invalid ARN prefix (Prefixo do ARN inválido)

Código do problema: INVALID_ARN_PREFIX

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Resolver o erro

Os ARNs de recursos da AWS devem incluir o prefixo `arn:` exigido.

Termos relacionados

- Recursos de políticas
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: Invalid ARN Region (Região do ARN inválida)

Código do problema: INVALID_ARN_REGION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Resolver o erro

Não há suporte para o tipo de recurso na região especificada. Para obter uma tabela dos produtos da AWS com suporte em cada região, consulte a Tabela de regiões.

Termos relacionados

- Recursos de políticas
- ARNs de recursos
- Nomes e códigos das regiões

Erro: Invalid ARN resource (Recurso do ARN inválido)

Código do problema: INVALID_ARN_RESOURCE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Resolver o erro

O ARN do recurso deve corresponder às especificações dos tipos de recursos conhecidos. Para visualizar o formato de ARN esperado para um serviço, consulteAções, recursos e chaves de

condição de produtos da AWS. Escolha o nome do serviço para visualizar seus tipos de recursos e formatos de ARN.

Termos relacionados

- Recursos de políticas
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: Invalid ARN service case (Caso de serviço de ARN inválido)

Código do problema: INVALID_ARN_SERVICE_CASE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid ARN service case: Update the service name {{service}} in the resource ARN to
 use all lowercase letters.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Update the service name {{service}} in the resource ARN to use all
 lowercase letters."
```

Resolver o erro

O serviço no ARN do recurso deve corresponder às especificações (incluindo a capitalização) dos prefixos de serviço. Para visualizar o prefixo de um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço e localize seu prefixo na primeira frase.

Termos relacionados

- Recursos de políticas
- ARNs de recursos

- AWS Recursos do produto da com formatos de ARN

Erro: Invalid condition data type (Tipo de dados da condição inválido)

Código do problema: INVALID_CONDITION_DATA_TYPE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Resolver o erro

O valor no par de chave-valor da condição deve corresponder ao tipo de dados da chave da condição e do operador da condição. Para visualizar o tipo de dados da chave da condição para um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço para visualizar as chaves de condição desse serviço.

Por exemplo, a chave de condição global `CurrentTime` oferece suporte ao operador de condição `Date`. Se você fornecer uma string ou um número inteiro para o valor no bloco de condição, o tipo de dados não corresponderá.

Termos relacionados

- Condições
- Bloco de condição
- Elementos de política JSON do IAM: operadores de condição
- Chaves de condições globais

- AWS Chaves de condição do produto da

Erro: Invalid condition key format (Formato de chave de condição inválido)

Código do problema: INVALID_CONDITION_KEY_FORMAT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Resolver o erro

A chave no par de chave-valor da condição deve corresponder às especificações do serviço. Para visualizar as chaves da condição de um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço para visualizar as chaves de condição desse serviço.

Termos relacionados

- Condições
- Chaves de condições globais
- AWS Chaves de condição do produto da

Erro: Invalid condition multiple Boolean (Condição inválida múltipla booliana)

Código do problema: INVALID_CONDITION_MULTIPLE_BOOLEAN

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Resolver o erro

A chave no par de chave-valor da condição espera um único valor booliano. Quando você fornece vários valores boolianos, a correspondência da condição pode não retornar os resultados esperados.

Para visualizar as chaves da condição de um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço para visualizar as chaves de condição desse serviço.

- Condições
- Chaves de condições globais
- AWS Chaves de condição do produto da

Erro: Invalid condition operator (Operador de condição inválido)

Código do problema: INVALID_CONDITION_OPERATOR

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Resolver o erro

Atualize a condição para usar um operador de condição com suporte.

Termos relacionados

- Elementos de política JSON do IAM: operadores de condição
- Elemento de condição
- Visão geral das políticas de JSON

Erro: Invalid effect (Efeito inválido)

Código do problema: INVALID_EFFECT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Resolver o erro

Atualize o elemento `Effect` para usar um efeito válido. Os valores válidos para `Effect` são **`Allow`** e **`Deny`**.

Termos relacionados

- Elemento Effect
- Visão geral das políticas de JSON

Erro: Invalid global condition key (Chave de condição global inválida)

Código do problema: INVALID_GLOBAL_CONDITION_KEY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Resolver o erro

Atualize a chave de condição no par de chave-valor da condição para usar uma chave de condição global com suporte.

As chaves de condição globais são chaves de condição com um prefixo `aws:`. Os serviços da AWS podem oferecer suporte a chaves de condição globais ou fornecer chaves específicas do serviço que incluem seu prefixo de serviço. Por exemplo, as chaves de condição do IAM incluem o prefixo `iam:`. Para obter mais informações, consulte Ações, recursos e chaves de condição de serviços da AWS e escolha o serviço cujas chaves você deseja visualizar.

Termos relacionados

- Chaves de condições globais

Erro: Invalid partition (Partição inválida)

Código do problema: INVALID_PARTITION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Resolver o erro

Atualize o ARN do recurso para incluir uma partição com suporte. Se você incluir uma partição com suporte, o serviço ou recurso poderá não oferecer suporte à partição incluída.

Uma partição é um grupo de regiões da AWS. Cada conta da AWS tem escopo para uma partição. Em regiões clássicas, use a partição `aws`. Nas regiões da China, use `aws-cn`.

Termos relacionados

- Nomes de recursos da Amazon (ARNs) - Partições

Erro: Invalid policy element (Elemento de política inválido)

Código do problema: INVALID_POLICY_ELEMENT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid policy element: The policy element {{element}} is not valid.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The policy element {{element}} is not valid."
```

Resolver o erro

Atualize a política para incluir apenas elementos de política JSON com suporte.

Termos relacionados

- Elementos de política JSON

Erro: Invalid principal format (Formato de entidade inválido)

Código do problema: INVALID_PRINCIPAL_FORMAT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid principal format: The Principal element contents are not valid. Specify a key-value pair in the Principal element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value pair in the Principal element."
```

Resolver o erro

Atualize a entidade para usar um formato de par de chave-valor com suporte.

É possível especificar uma entidade principal em uma política baseada em recurso, mas não em uma política baseada em identidade.

Por exemplo, para definir o acesso para todos em uma conta da AWS, use a seguinte entidade em sua política:

```
"Principal": { "AWS": "123456789012" }
```

Termos relacionados

- Elementos de política JSON: entidade
- Políticas baseadas em identidade e em recurso

Erro: Invalid principal key (Chave de entidade inválida)

Código do problema: INVALID_PRINCIPAL_KEY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Resolver o erro

Atualize a chave no par de chave-valor da entidade para usar uma chave de entidade com suporte. As seguintes chaves de entidade com suporte são:

- AWS
- CanonicalUser
- Federado
- Serviço

Termos relacionados

- Elemento Principal

Erro: Invalid Region (Região inválida)

Código do problema: INVALID_REGION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 supported Region.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 supported Region."
```

Resolver o erro

Atualize o valor do par da chave-valor da condição para incluir uma região com suporte. Para obter uma tabela dos produtos da AWS com suporte em cada região, consulte a Tabela de regiões.

Termos relacionados

- Recursos de políticas
- ARNs de recursos
- Nomes e códigos das regiões

Erro: Invalid service (Serviço inválido)

Código do problema: INVALID_SERVICE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Resolver o erro

O prefixo de serviço na chave de ação ou condição deve corresponder às especificações (incluindo a capitalização) dos prefixos de serviço. Para visualizar o prefixo de um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço e localize seu prefixo na primeira frase.

Termos relacionados

- Serviços conhecidos e suas ações, recursos e chaves de condição

Erro: Invalid service condition key (Chave de condição de serviço inválida)

Código do problema: INVALID_SERVICE_CONDITION_KEY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Resolver o erro

Atualize a chave no par de chave-valor da condição para usar uma chave de condição conhecida para o serviço. Os nomes das chaves de condições globais começam com o prefixo `aws`.
Os produtos da AWS podem fornecer chaves específicas de serviço que incluem o prefixo correspondente do serviço. Para visualizar o prefixo de um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS.

Termos relacionados

- Chaves de condições globais
- Serviços conhecidos e suas ações, recursos e chaves de condição

Erro: Invalid service in action (Serviço inválido na ação)

Código do problema: INVALID_SERVICE_IN_ACTION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Resolver o erro

O prefixo de serviço na ação deve corresponder às especificações (incluindo a capitalização) dos prefixos de serviço. Para visualizar o prefixo de um serviço, consulte Ações, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço e localize seu prefixo na primeira frase.

Termos relacionados

- Elemento Action
- Serviços conhecidos e suas ações

Erro: Invalid variable for operator (Variável inválida para o operador)

Código do problema: INVALID_VARIABLE_FOR_OPERATOR

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Resolver o erro

É possível usar variáveis de política no elemento `Resource` e em comparações de string no elemento `Condition`. As condições oferecem suporte a variáveis quando você usa operadores de string ou operadores de ARN. Os operadores de string incluem `StringEquals`, `StringLike` e `StringNotLike`. Os operadores de ARN incluem `ArnEquals` e `ArnLike`. Não é possível usar uma variável de política com outros operadores, como operadores do tipo numérico, de data, booliano, binário, de endereço IP ou nulo.

Termos relacionados

- Uso de variáveis de política no elemento Condition
- Elemento de condição

Erro: Invalid version (Versão inválida)

Código do problema: INVALID_VERSION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid version: The version {{version}} is not valid. Use one of the following
 versions: {{versions}}
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The version {{version}} is not valid. Use one of the following
 versions: {{versions}}"
```

Resolver o erro

O elemento de política `Version` especifica as regras de sintaxe de linguagem que a AWS deve usar para processar uma política. Para usar todos os recursos de política disponíveis, inclua o elemento `Version` mais recente antes do elemento `Statement` em todas as suas políticas.

```
"Version": "2012-10-17"
```

Termos relacionados

- Elemento Version

Erro: Json syntax error (Erro de sintaxe Json)

Código do problema: JSON_SYNTAX_ERROR

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Resolver o erro

Sua política inclui um erro de sintaxe. Verifique sua sintaxe JSON.

Termos relacionados

- Validador JSON
- Referência de elementos de política JSON do IAM
- Visão geral das políticas de JSON

Erro: Json syntax error (Erro de sintaxe Json)

Código do problema: JSON_SYNTAX_ERROR

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Json syntax error: Fix the JSON syntax error.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Fix the JSON syntax error."
```

Resolver o erro

Sua política inclui um erro de sintaxe. Verifique sua sintaxe JSON.

Termos relacionados

- Validador JSON
- Referência de elementos de política JSON do IAM
- Visão geral das políticas de JSON

Erro: Missing action (Ação ausente)

Código do problema: MISSING_ACTION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing action: Add an Action or NotAction element to the policy statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Resolver o erro

As políticas de JSON da AWS devem incluir um elemento `Action` ou `NotAction`.

Termos relacionados

- Elemento Action
- Elemento NotAction
- Visão geral das políticas de JSON

Erro: Missing ARN field (Campo de ARN ausente)

Código do problema: MISSING_ARN_FIELD

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Resolver o erro

Todos os campos no ARN do recurso devem corresponder às especificações de um tipo de recurso conhecido. Para visualizar o formato de ARN esperado para um serviço, consulteAções, recursos e chaves de condição de produtos da AWS. Escolha o nome do serviço para visualizar seus tipos de recursos e formatos de ARN.

Termos relacionados

- Recursos de políticas
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: Missing ARN Region (Região do ARN ausente)

Código do problema: MISSING_ARN_REGION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Resolver o erro

Os ARNs de recursos para a maioriados produtos da AWS exigem a especificação de uma região. Para obter uma tabela dos produtos da AWS com suporte em cada região, consulte a Tabela de regiões.

Termos relacionados

- Recursos de políticas
- ARNs de recursos
- Nomes e códigos das regiões

Erro: Missing effect (Efeito ausente)

Código do problema: MISSING_EFFECT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Resolver o erro

As políticas de JSON da AWS devem incluir um elemento `Effect` com o valor **`Allow`** e **`Deny`**.

Termos relacionados

- Elemento Effect
- Visão geral das políticas de JSON

Erro: Missing principal (Entidade ausente)

Código do problema: MISSING_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing principal: Add a Principal element to the policy statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a Principal element to the policy statement."
```

Resolver o erro

As políticas baseadas em recursos devem incluir um elemento `Principal`.

Por exemplo, para definir o acesso para todos em uma conta da AWS, use a seguinte entidade em sua política:

```
"Principal": { "AWS": "123456789012" }
```

Termos relacionados

- Elemento Principal
- Políticas baseadas em identidade e em recurso

Erro:Missing qualifier (Qualificador ausente)

Código do problema: MISSING_QUALIFIER

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing qualifier: The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The request context key {{key}} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Resolver o erro

No elemento `Condition`, crie expressões em que você use operadores de condição, como igual ou menor que para comparar uma condição na política com chaves e valores no contexto da solicitação. Para solicitações que incluem vários valores para uma única chave de condição, você deve colocar as condições entre colchetes como uma matriz ("Key2":["Value2A", "Value2B"]). Também é necessário usar os operadores de conjunto `ForAllValues` ou `ForAnyValue` com o operador de condição `StringLike`. Esses qualificadores adicionam a funcionalidade de operação de conjuntos ao operador da condição, para que você possa testar vários valores de solicitação em relação a vários valores de condição.

Termos relacionados

- Chaves de contexto de múltiplos valores
- Elemento de condição

AWSPolíticas gerenciadas pela com esse erro

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

As políticas gerenciadas pela AWS a seguir incluem um qualificador ausente para chaves de condição em suas instruções de política. Ao usar a política gerenciada pela AWS como referência para criar sua política gerenciada pelo cliente, a AWS recomenda que você adicione os qualificadores de chave de condição `ForAllValues` ou `ForAnyValue` ao seu elemento `Condition`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Erro: Missing resource (Recurso ausente)

Código do problema: MISSING_RESOURCE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Resolver o erro

Todas as políticas, exceto as políticas de confiança de perfil, devem incluir um elemento `Resource` ou `NotResource`.

Termos relacionados

- Elemento de recurso
- Elemento NotResource
- Políticas baseadas em identidade e em recurso
- Visão geral das políticas de JSON

Erro: Missing statement (Instrução ausente)

Código do problema: MISSING_STATEMENT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing statement: Add a statement to the policy
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a statement to the policy"
```

Resolver o erro

Uma política JSON deve incluir uma instrução.

Termos relacionados

- Elementos de política JSON

Erro: Null with if exists (Nulo com IfExists)

Código do problema: NULL_WITH_IF_EXISTS

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Resolver o erro

É possível adicionar `IfExists` ao final de qualquer nome de operador de condição, exceto o operador de condição `Null`. Use um operador de condição `Null` para verificar se uma chave de condição está presente no momento da autorização. Use `...ifExists` para dizer “Se a chave de política estiver presente no contexto da solicitação, processar a chave conforme especificado na política. Se a chave não estiver presente, avalie o elemento da condição como verdadeiro."

Termos relacionados

- ...Operadores de condição ...IfExists
- Operador de condição Null
- Elemento de condição

Erro: SCP syntax error action wildcard (Erro de sintaxe de SCP na ação com curinga)

Código do problema: SCP_SYNTAX_ERROR_ACTION_WILDCARD

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations oferecem suporte à especificação de valores nos elementos `Action` ou `NotAction`. No entanto, esses valores podem incluir caracteres curinga (*) somente no final da string. Isso significa que você pode especificar `iam:Get*`, mas não `iam:*role`.

Para especificar várias ações, a AWS recomenda que você as liste individualmente.

Termos relacionados

- Ação de SCP e elementos NotAction
- Avaliação do SCP
- AWS Organizations Políticas de controle de serviço do
- Elementos de política JSON do IAM: Action

Erro: SCP syntax error principal (Erro de sintaxe de SCP para a entidade)

Código do problema: SCP_SYNTAX_ERROR_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations não oferecem suporte aos elementos `Principal` ou `NotPrincipal`.

É possível especificar o nome do recurso da Amazon (ARN) usando a chave de condição global `aws:PrincipalArn` no elemento `Condition`.

Termos relacionados

- Sintaxe de SCP
- Chaves de condição globais para entidades principais

Erro: Unique Sids required (Sids exclusivos obrigatórios)

Código do problema: UNIQUE_SIDS_REQUIRED

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Resolver o erro

Para alguns tipos de política, os IDs de instrução devem ser exclusivos. O elemento `Sid` (ID de instrução) permite inserir um identificador opcional fornecido para a instrução de política. É possível atribuir um valor de ID de instrução a cada instrução em uma matriz de instruções usando o elemento `SID`. Em serviços que permitem que você especifique um elemento de ID, como o SQS e o SNS, o valor `Sid` é apenas um subID do ID do documento de política. Por exemplo, no IAM, o valor `Sid` deve ser exclusivo em uma política JSON.

Termos relacionados

- Elementos de política JSON do IAM: Sid

Erro: ação não suportada na política

Código do problema: UNSUPPORTED_ACTION_IN_POLICY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver o erro

Algumas ações não são compatíveis com o elemento `Action` na política baseada em recursos anexada a um tipo de recurso diferente. Por exemplo, as ações do AWS Key Management Service não são compatíveis com políticas de bucket do Amazon S3. Especifique uma ação compatível com o tipo de recurso anexado à sua política baseada em recursos.

Termos relacionados

- Elementos de política JSON: Action

Erro: Unsupported element combination (Combinação de elementos não suportada)

Código do problema: UNSUPPORTED_ELEMENT_COMBINATION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported element combination: The policy elements {{element1}} and {{element2}} can
 not be used in the same statement. Remove one of these elements.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The policy elements {{element1}} and {{element2}} can not be used in
 the same statement. Remove one of these elements."
```

Resolver o erro

Algumas combinações de elementos de política JSON não podem ser usadas juntas. Por exemplo, você não pode usar `Action` e `NotAction` na mesma instrução de política. Outros pares que são mutuamente exclusivos incluem `Principal/NotPrincipal` e `Resource/NotResource`.

Termos relacionados

- Referência de elementos de política JSON do IAM
- Visão geral das políticas de JSON

Erro: Unsupported global condition key (Chave de condição global incompatível)

Código do problema: UNSUPPORTED_GLOBAL_CONDITION_KEY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Resolver o erro

AWSA não oferece suporte ao uso da chave de condição global especificada. Dependendo do seu caso de uso, você pode usar as chaves de condição globais `aws:PrincipalArn` ou `aws:SourceArn`. Por exemplo, em vez de `aws:ARN`, use `aws:PrincipalArn` para comparar o nome do recurso da Amazon (ARN) da entidade que fez a solicitação com o ARN especificado na política. Como alternativa, use a chave de condição global `aws:SourceArn` para comparar o nome do recurso da Amazon (ARN) do recurso que está fazendo uma solicitação de serviço a serviço com o ARN especificado na política.

Termos relacionados

- AWS Chaves contextuais de condições globais da

Erro: Unsupported principal (Sem suporte para a entidade)

Código do problema: UNSUPPORTED_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported principal: The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The policy type {{policy_type}} does not support the Principal
 element. Remove the Principal element."
```

Resolver o erro

O elemento `Principal` especifica a entidade que tem acesso permitido ou negado a um recurso. Você não pode usar o elemento `Principal` em uma política baseada em identidade do IAM. É possível usá-lo nas políticas de confiança para funções do IAM e em políticas baseadas em recurso. As políticas baseadas em recursos são políticas que você incorpora diretamente em um recurso. Por exemplo, você pode incorporar políticas em um bucket do Amazon S3 ou uma chave do AWS KMS.

Termos relacionados

- AWS Elementos de política JSON da : entidade
- Acesso a recursos entre contas no IAM

Erro: ARN de recurso incompatível na política

Código do problema: UNSUPPORTED_RESOURCE_ARN_IN_POLICY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver o erro

Alguns ARNs de recurso são incompatíveis com o elemento `Resource` da política baseada em recurso quando a política é anexada a um tipo de recurso diferente. Por exemplo, os ARNs do AWS KMS são incompatíveis com o elemento `Resource` das políticas de bucket do Amazon S3.

Especifique um ARN de recurso compatível com o tipo de recurso anexado à sua política baseada em recursos.

Termos relacionados

- Elementos de política JSON: Action

Erro: Unsupported Sid (Não há suporte para o Sid)

Código do problema: UNSUPPORTED_SID

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported Sid: Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Resolver o erro

O elemento `Sid` é compatível com letras maiúsculas, letras minúsculas e números.

Termos relacionados

- Elementos de política JSON do IAM: Sid

Erro: Unsupported wildcard in principal (Não há suporte para o curinga na entidade)

Código do problema: UNSUPPORTED_WILDCARD_IN_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Resolver o erro

A estrutura do elemento `Principal` oferece suporte ao uso de um par de chave-valor. O valor da entidade especificado na política inclui um caractere curinga (*). Não é possível incluir um caractere curinga com a chave da entidade especificada. Por exemplo, ao especificar usuários em um elemento `Principal`, você não pode usar um caractere curinga para se referir a “todos os usuários”. Você deve indicar um usuário ou usuários específicos. Da mesma forma, ao especificar uma sessão de função assumida, você não pode usar um caractere curinga para se referir a “todas as sessões”. Você deve indicar uma sessão específica. Não é possível usar um caractere curinga para corresponder a parte de um nome ou um ARN.

Para resolver essa descoberta, remova o caractere curinga e forneça uma entidade mais específica.

Termos relacionados

- AWS Elementos de política JSON da : entidade

Erro: Missing brace in variable (Chave ausente na variável)

Código do problema: MISSING_BRACE_IN_VARIABLE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Resolver o erro

A estrutura de variável de política suporta o uso de um prefixo $seguido por um par de chaves ({ }). Dentro dos caracteres ${ }, inclua o nome do valor da solicitação que você deseja usar na política.

Para resolver essa descoberta, adicione a chave ausente para garantir que o conjunto completo de chaves de abertura e fechamento esteja presente.

Termos relacionados

- Elementos de política do IAM: variáveis

Erro: Missing quote in variable (Aspa ausente na variável)

Código do problema: MISSING_QUOTE_IN_VARIABLE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Resolver o erro

Ao adicionar uma variável à sua política, você pode especificar um valor padrão para a variável. Se uma variável não estiver presente, a AWS usa o texto padrão fornecido por você.

Para adicionar um valor padrão a uma variável, coloque o valor padrão entre aspas simples (`' '`) e separe o texto da variável e o valor padrão com uma vírgula e espaço (`, `).

Por exemplo, se uma entidade principal estiver marcada com `team=yellow`, eles podem acessar o bucket `amzn-s3-demo-bucket` do Amazon S3 de nome `amzn-s3-demo-bucket-yellow`. É possível que uma política com esse recurso permita que os membros da equipe acessem seus próprios recursos, mas não os de outras equipes. Para usuários sem etiquetas de equipe, você pode configurar um valor padrão de `company-wide`. Esses usuários podem acessar somente o bucket `amzn-s3-demo-bucket-company-wide`, onde podem visualizar informações gerais, como instruções para ingressar em uma equipe.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Termos relacionados

- Elementos de política do IAM: variáveis

Erro: Unsupported space in variable Não há suporte para espaço na variável)

Código do problema: UNSUPPORTED_SPACE_IN_VARIABLE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Resolver o erro

A estrutura de variável de política suporta o uso de um prefixo $seguido por um par de chaves (`{ }`). Dentro dos caracteres `${ }`, inclua o nome do valor da solicitação que você deseja usar na política.

Embora você possa incluir um espaço ao especificar uma variável padrão, não é possível incluir um espaço no nome da variável.

Termos relacionados

- Elementos de política do IAM: variáveis

Erro: Empty variable (Variável vazia)

Código do problema: EMPTY_VARIABLE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Resolver o erro

A estrutura de variável de política suporta o uso de um prefixo $seguido por um par de chaves ({ }). Dentro dos caracteres ${ }, inclua o nome do valor da solicitação que você deseja usar na política.

Termos relacionados

- Elementos de política do IAM: variáveis

Erro: Variable unsupported in element (Não há suporte para a variável no elemento)

Código do problema: VARIABLE_UNSUPPORTED_IN_ELEMENT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Resolver o erro

É possível usar variáveis de política no elemento `Resource` e em comparações de string no elemento `Condition`.

Termos relacionados

- Elementos de política do IAM: variáveis

Erro: Variable unsupported in version (Não há suporte para a variável na versão)

Código do problema: VARIABLE_UNSUPPORTED_IN_VERSION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Resolver o erro

Para usar variáveis de política, você deve incluir o elemento `Version` e configurá-lo para uma versão que ofereça suporte a variáveis de política. As variáveis foram apresentadas na versão `2012-10-17`. Versões anteriores da linguagem da política não são compatíveis com variáveis de política. Se você não definir a `Version` como `2012-10-17` ou posterior, variáveis como `${aws:username}` serão tratadas como strings literais na política.

Um elemento de política `Version` é diferente de uma versão de política. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. Uma versão da política é criada quando você altera uma política gerenciada pelo cliente no IAM. A política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada.

Termos relacionados

- Elementos de política do IAM: variáveis
- Elementos de política JSON do IAM: Version

Erro: Private IP address (Endereço IP privado)

Código do problema: PRIVATE_IP_ADDRESS

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Resolver o erro

A chave de condição global `aws:SourceIp` funciona apenas para intervalos de endereços IP públicos. Você recebe esse erro quando sua política permite apenas endereços IP privados. Nesse caso, a condição nunca corresponderá.

- Chave de condição global aws:SourceIp
- Elementos de política JSON do IAM: Condition

Erro: Private NotIpAddress (NotipAddress privado)

Código do problema: PRIVATE_NOT_IP_ADDRESS

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Resolver o erro

A chave de condição global `aws:SourceIp` funciona apenas para intervalos de endereços IP públicos. Você recebe esse erro quando usa o operador de condição `NotIpAddress` e lista apenas endereços IP privados. Nesse caso, a condição sempre corresponderá e será ineficaz.

- Chave de condição global aws:SourceIp
- Elementos de política JSON do IAM: Condition

Erro: Policy size exceeds SCP quota (O tamanho da política excede a cota de SCP)

Código do problema: POLICY_SIZE_EXCEEDS_SCP_QUOTA

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations oferecem suporte à especificação de valores nos elementos `Action` ou `NotAction`. No entanto, esses valores podem incluir caracteres curinga (*) somente no final da string. Isso significa que você pode especificar `iam:Get*`, mas não `iam:*role`.

Para especificar várias ações, a AWS recomenda que você as liste individualmente.

Termos relacionados

- Cotas do AWS Organizations
- AWS Organizations Políticas de controle de serviço do

Erro: Invalid service principal format (Formato de entidade de serviço inválido)

Código do problema: INVALID_SERVICE_PRINCIPAL_FORMAT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Resolver o erro

O valor no par de chave-valor da condição deve corresponder a um formato de entidade de serviço definido.

Um escopo principal do serviço é um identificador que é usado para conceder permissões a um serviço. É possível especificar uma entidade principal de serviço no elemento `Principal` ou como um valor para algumas chaves de condição globais e chaves específicas do serviço. A entidade do serviço é definida por cada serviço.

O identificador de uma entidade principal de serviço inclui o nome do serviço e geralmente está todo em letras minúsculas, no seguinte formato:

`service-name``.amazonaws.com`

Algumas chaves específicas do serviço podem usar um formato diferente para entidades principais de serviço. Por exemplo, a chave da condição `kms:ViaService` requer o seguinte formato para entidades principais de serviço em letras minúsculas:

`service-name.AWS_region``.amazonaws.com`

Termos relacionados

- Entidades principais de serviço
- AWS Chaves de condições globais da
- `kms:ViaService` Chave da condição

Erro: Missing tag key in condition (Chave de etiqueta ausente na condição)

Código do problema: MISSING_TAG_KEY_IN_CONDITION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Resolver o erro

Para controlar o acesso com base em tags, forneça informações sobre a tag no elemento de condição de uma política.

Por exemplo, para controlar o acesso aos recursos da AWS, inclua a chave de condição `aws:ResourceTag`. Esta chave requer o formato `aws:ResourceTag/`*`tag-key`*. Para especificar a chave de etiqueta `owner` e o valor de etiqueta `JaneDoe` em uma condição, use o formato a seguir.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

Termos relacionados

- Controlar o acesso usando tags
- Condições
- Chaves de condições globais
- AWS Chaves de condição do produto da

Erro: formato de vpc inválido

Código do problema: INVALID_VPC_FORMAT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Resolver o erro

A chave de condição `aws:SourceVpc` deve usar o prefixo `vpc-` seguido por 8 ou 17 caracteres alfanuméricos, por exemplo, `vpc-11223344556677889` ou `vpc-12345678`.

Termos relacionados

- AWS Chaves globais de condição da : aws:SourceVpc

Erro: formato de vpce inválido

Código do problema: INVALID_VPCE_FORMAT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Resolver o erro

A chave de condição `aws:SourceVpce` deve usar o prefixo `vpce-` seguido por 8 ou 17 caracteres alfanuméricos, por exemplo, `vpce-11223344556677889` ou `vpce-12345678`.

Termos relacionados

- AWS Chaves globais de condição da : aws:SourceVpce

Erro: a entidade principal federada é incompatível

Código do problema: FEDERATED_PRINCIPAL_NOT_SUPPORTED

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Resolver o erro

O elemento `Principal` usa entidades principais federadas para políticas de confiança anexadas a funções do IAM a fim de fornecer acesso por meio de federação de identidades. Políticas de identidade e outras políticas baseadas em recursos são incompatíveis com um provedor de identidade federado no elemento `Principal`. Por exemplo, você não pode usar uma entidade principal SAML em uma política de bucket do Amazon S3. Modifique o elemento `Principal` para um tipo de entidade principal compatível.

Termos relacionados

- Criar uma função para federação de identidades
- Elementos de política JSON: entidade principal

Erro: ação incompatível para a chave de condição

Código do problema: UNSUPPORTED_ACTION_FOR_CONDITION_KEY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
 recommend that you move these actions to a different statement without this condition
 key."
```

Resolver o erro

Verifique se a chave de condição no elemento `Condition` da declaração de política é aplicável a todas as ações do elemento `Action`. Para garantir que as ações especificadas sejam efetivamente permitidas ou negadas por sua política, você deve mover as ações incompatíveis para uma declaração diferente sem a chave de condição.

> **Note**
>
> Se o elemento `Action` tiver ações com curingas, o IAM Access Analyzer não avalia essas ações para esse erro.

Termos relacionados

- Elementos de política JSON: Action

Erro: ação não suportada na política

Código do problema: UNSUPPORTED_ACTION_IN_POLICY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver o erro

Algumas ações não são compatíveis com o elemento `Action` na política baseada em recursos anexada a um tipo de recurso diferente. Por exemplo, as ações do AWS Key Management Service não são compatíveis com políticas de bucket do Amazon S3. Especifique uma ação compatível com o tipo de recurso anexado à sua política baseada em recursos.

Termos relacionados

- Elementos de política JSON: Action

Erro: ARN de recurso incompatível na política

Código do problema: UNSUPPORTED_RESOURCE_ARN_IN_POLICY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver o erro

Alguns ARNs de recurso são incompatíveis com o elemento `Resource` da política baseada em recurso quando a política é anexada a um tipo de recurso diferente. Por exemplo, os ARNs do AWS KMS são incompatíveis com o elemento `Resource` das políticas de bucket do Amazon S3. Especifique um ARN de recurso compatível com o tipo de recurso anexado à sua política baseada em recursos.

Termos relacionados

- Elementos de política JSON: Action

Erro: chave de condição incompatível para a entidade principal de serviço

Código do problema: UNSUPPORTED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Resolver o erro

É possível especificar Serviços da AWS no elemento `Principal` de uma política baseada em recursos usando uma entidade principal de serviço, que é um identificador desse serviço. Não é possível usar algumas chaves de condição com determinadas entidades principais de serviço. Por exemplo, você não pode usar a chave de condição `aws:PrincipalOrgID` com a entidade principal de serviço `cloudfront.amazonaws.com`. É necessário remover as chaves de condição que não são aplicáveis à entidade principal de serviço no elemento `Principal`.

Termos relacionados

- Entidades principais de serviço
- Elementos de política JSON: entidade principal

Erro: Role trust policy syntax error notprincipal (Erro de sintaxe da política de confiança da função notprincipal)

Código do problema: ROLE_TRUST_POLICY_SYNTAX_ERROR_NOTPRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Resolver o erro

Uma política de confiança da função é uma política baseada no recurso que é anexada a um perfil do IAM. As políticas de confiança definem quais entidades principais (contas, usuários, funções e usuários federados) podem assumir a função. As políticas de confiança das funções não são compatíveis com `NotPrincipal`. Atualize a política para usar um elemento `Principal` em vez disso.

Termos relacionados

- Elementos de política JSON: entidade
- Elementos de política JSON: NotPrincipal

Erro: Role trust policy unsupported wildcard in principal (Curinga incompatível da política de confiança da ação na entidade principal)

Código do problema: ROLE_TRUST_POLICY_UNSUPPORTED_WILDCARD_IN_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Resolver o erro

Uma política de confiança da função é uma política baseada no recurso que é anexada a um perfil do IAM. As políticas de confiança definem quais entidades principais (contas, usuários, funções e usuários federados) podem assumir a função. `"Principal:" "*"` não é compatível com o elemento `Principal` de uma política de confiança da função. Substitua o curinga por um valor de entidade principal válido.

Termos relacionados

- Elementos de política JSON: entidade

Erro: Role trust policy syntax error resource (Erro de sintaxe da política de confiança da função resource)

Código do problema: ROLE_TRUST_POLICY_SYNTAX_ERROR_RESOURCE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Resolver o erro

Uma política de confiança da função é uma política baseada no recurso que é anexada a um perfil do IAM. As políticas de confiança definem quais entidades principais (contas, usuários, funções e usuários federados) podem assumir a função. As políticas de confiança das funções se aplicam à função à qual estão anexadas. Não é possível especificar um elemento `Resource` ou `NotResource` em uma política de confiança da função. Remover o elemento `Resource` ou `NotResource`.

- Elementos de política JSON: Resource
- Elementos de política JSON: NotResource

Erro: intervalo IP de incompatibilidade de tipos

Código do problema: TYPE_MISMATCH_IP_RANGE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Resolver o erro

Atualize o texto para usar o tipo de dados do operador de condição de endereço IP, em um formato de CIDR.

Termos relacionados

- Operadores de condição de endereço IP
- Elementos de política JSON do IAM: operadores de condição

Erro: Missing action for condition key (Ação ausente para chave de condição)

Código do problema: MISSING_ACTION_FOR_CONDITION_KEY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Resolver o erro

A chave de condição no elemento `Condition` da instrução da política não é avaliado, a menos que a ação especificada esteja no elemento `Action`. Para garantir que as chaves de condição especificadas sejam de fato permitidas ou negadas pela política, adicione a ação ao elemento `Action`.

Termos relacionados

- Elementos de política JSON: Action

Erro: Invalid federated principal syntax in role trust policy (Sintaxe de entidade principal federada inválida na política de confiança da função)

Código do problema: INVALID_FEDERATED_PRINCIPAL_SYNTAX_IN_ROLE_TRUST_POLICY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Resolver o erro

O valor de entidade principal especifica uma entidade principal federada que não corresponde ao formato esperado. Atualize o formato da entidade principal federada para um nome de domínio válido ou um ARN de metadados SAML.

Termos relacionados

- Usuários federados e funções

Erro: Mismatched action for principal (Ação incompatível com a entidade principal)

Código do problema: MISMATCHED_ACTION_FOR_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Resolver o erro

A ação especificada no elemento `Action` da instrução da política é inválido com a entidade principal especificada no elemento `Principal`. Por exemplo, você não pode usar uma entidade principal do provedor SAML com a ação `sts:AssumeRoleWithWebIdentity`. Você deve usar uma entidade

principal de provedor SAML com a ação `sts:AssumeRoleWithSAML` ou usar uma entidade principal de provedor do OIDC com a ação `sts:AssumeRoleWithWebIdentity`.

Termos relacionados

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

Erro: Missing action for roles anywhere trust policy (Ação ausente para política de confiança do roles anywhere)

Código do problema: MISSING_ACTION_FOR_ROLES_ANYWHERE_TRUST_POLICY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Resolver o erro

Para que o IAM Roles Anywhere possa assumir uma função e fornecer credenciais temporários da AWS, a função deve confiar na entidade principal do serviço. A entidade principal do serviço IAM Roles Anywhere exige as permissões de `sts:AssumeRole`, `sts:SetSourceIdentity` e `sts:TagSession` para assumir uma função. Se qualquer uma das permissões estiver ausente, você deve adicioná-la à política.

Termos relacionados

- Modelo de confiança do AWS Identity and Access Management Roles Anywhere

Erro: o tamanho da política excede a cota de RCP

Código do problema: POLICY_SIZE_EXCEEDS_RCP_QUOTA

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Policy size exceeds RCP quota: The {{policySize}} characters in the resource control
 policy (RCP) exceed the {{policySizeQuota}} character maximum for RCPs. We recommend
 that you use multiple granular policies.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{policySize}} characters in the resource control policy (RCP)
 exceed the {{policySizeQuota}} character maximum for RCPs. We recommend that you use
 multiple granular policies."
```

Resolver o erro

As políticas de controle de recurso (RCPs) do AWS Organizations oferecem suporte à especificação de valores no elemento `Action`. No entanto, esses valores podem incluir caracteres curinga (*) somente no final da string. Isso significa que você pode especificar `s3:Get*`, mas não `s3:*Object`.

Para especificar várias ações, a AWS recomenda que você as liste individualmente.

Termos relacionados

- Cotas para o AWS Organizations
- AWS Organizations Políticas de controle de recursos do

Erro: erro de sintaxe de RCP para entidade principal

Código do problema: RCP_SYNTAX_ERROR_PRINCIPAL

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP syntax error principal: The Principal element contents are not valid. RCPs only
 support specifying all principals ("*") in the Principal element. The NotPrincipal
 element is not supported for RCPs.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Principal element contents are not valid. RCPs only support
 specifying all principals ("*") in the Principal element. The NotPrincipal element is
 not supported for RCPs."
```

Resolver o erro

As políticas de controle de recurso (RCPs) do AWS Organizations oferecem suporte somente à especificação de todas as entidades principais ("*") no elemento `Principal`. Não há suporte para o elemento `NotPrincipal` nas RCPs.

Termos relacionados

- Sintaxe de RCP
- Propriedades da entidade principal

Erro: erro de sintaxe de RCP ao permitir

Código do problema: RCP_SYNTAX_ERROR_ALLOW

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP syntax error allow: RCPs only support specifying all principals ("*") in the
 Principal element, all resources ("*") in the Resource element, and no Condition
 element with an effect of Allow.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "RCPs only support specifying all principals ("*") in the Principal
 element, all resources ("*") in the Resource element, and no Condition element with an
 effect of Allow."
```

Resolver o erro

As políticas de controle de recurso (RCPs) do AWS Organizations oferecem suporte somente à especificação de todas as entidades principais ("*") no elemento `Principal` e todos os recursos ("*") no elemento `Resource`. Não há suporte para o elemento `Condition` com um efeito de `Allow` nas RCPs.

Termos relacionados

- Sintaxe de RCP
- Propriedades da entidade principal
- Propriedades do recurso

Error: erro de sintaxe de RCP NotAction

Código do problema: RCP_SYNTAX_ERROR_NOTACTION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP syntax error NotAction: RCPs do not support the NotAction element. Update to use
 the Action element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "RCPs do not support the NotAction element. Update to use the Action
 element."
```

Resolver o erro

As políticas de controle de recurso (RCPs) do AWS Organizations não oferecem suporte ao elemento `NotAction`. Use o elemento `Action`.

Termos relacionados

- Sintaxe de RCP
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: NotAction

Erro: erro de sintaxe de RCP em ação

Código do problema: RCP_SYNTAX_ERROR_ACTION

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP syntax error action: RCPs only support specifying select service prefixes in the
 Action element. Learn more here.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "RCPs only support specifying select service prefixes in the Action
 element. Learn more here."
```

Resolver o erro

As políticas de controle de recurso (RCPs) do AWS Organizations oferecem suporte somente à especificação de prefixos de serviço de seleção no elemento `Action`.

Termos relacionados

- Sintaxe de RCP
- Lista de Serviços da AWS com suporte para RCPs

Erro: conta do ARN faltando

Código do problema: MISSING_ARN_ACCOUNT

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing ARN account: The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The resource {{resourceName}} in the arn is missing an account id.
 Please provide a 12 digit account id."
```

Resolver o erro

Inclua uma ID de conta no ARN do recurso. Os IDs de conta são números inteiros de 12 dígitos. Para saber como visualizar o ID da conta, consulte Localizar o ID da conta da AWS.

Termos relacionados

- Recursos de políticas
- Identificadores de conta
- ARNs de recursos
- AWS Recursos do produto da com formatos de ARN

Erro: valor de chave do kms inválido

Código de problema: INVALID_KMS_KEY_VALUE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid kms key value: The {{key}} condition key value must be a valid KMS key ARN.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{key}} condition key value must be a valid KMS key ARN."
```

Resolver o erro

Um ARN (Amazon Resource Name) AWS KMS key é um identificador exclusivo e totalmente qualificado para uma chave do KMS. Um ARN de chave inclui a Conta da AWS, a região e o ID de chave. Uma chave de ARN segue este formato:

`arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*

Termos relacionados

- Encontrar o ID e o ARN da chave
- ARN de chave
- AWS Chaves de contexto de condições globais da

Erro: uso muito permissivo de variáveis

Código do problema: VARIABLE_USAGE_TOO_PERMISSIVE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Variable usage too permissive: Overly permissive use of policy variable for the {{key}}
 condition key. Use the policy variable preceded by 6 consecutive characters.
```

```
The policy variable is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and policy variables can be evaluated as effective wildcards. Therefore
 policy variables are not allowed to be used with sensitive keys. Refer to https://
docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html for the
 list of sensitive keys.
```

```
Overly permissive use of policy variable with aws:userID. Use the policy variable on
 the right side of colon or as the only character on the left side of a colon.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Overly permissive use of policy variable for the {{key}} condition
 key. Use the policy variable preceded by 6 consecutive characters."
```

```
"findingDetails": "The policy variable is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and policy variables can be evaluated as
 effective wildcards. Therefore policy variables are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of policy variable with aws:userID. Use the
 policy variable on the right side of colon or as the only character on the left side
 of a colon."
```

Resolver o erro

Há três mensagens de descoberta possíveis para esse erro.

- Para a primeira mensagem de erro, modifique o uso de variáveis de política para ser mais específico. Adicione pelo menos seis caracteres consecutivos antes da variável de política para reduzir o escopo das permissões. Por exemplo, em vez de usar `${aws:username}`, use `prefix-${aws:username}` ou `myapp-${aws:username}`. Isso garante que a variável de política não conceda um acesso amplo demais.
- Para a segunda mensagem de erro, remova a variável de política da chave de condição especificada. As variáveis de políticas podem funcionar como curingas eficazes e não são permitidas com chaves de condição confidenciais por motivos de segurança. Em vez disso, use valores estáticos específicos ou considere reestruturar a política para usar chaves de condição não confidenciais que sejam compatíveis com variáveis de políticas.
- Para a primeira mensagem de erro, modifique o uso da variável de política `aws:userID` para ser mais restritivo. Coloque a variável de política do lado direito do sinal dois-pontos (depois do ID da conta) ou use-a como o único caractere do lado esquerdo do sinal de dois-pontos. Por exemplo, use `AIDACKCEVSQ6C2EXAMPLE:${aws:userid}` ou `${aws:userid}:*` vez de `${aws:userid}`.

Termos relacionados

- Elementos da política do IAM: variáveis e tags
- Elementos da política do IAM: condição
- Condições com várias chaves ou valores de contexto
- AWS Chaves de contexto de condições globais da

Erro: uso muito permissivo de curingas

Código do problema:WILDCARD_USAGE_TOO_PERMISSIVE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Wildcard usage too permissive: Overly permissive use of wildcard for the {{key}}
 condition key. Use the wildcard preceded by 6 consecutive characters.
```

```
The wildcard is not allowed in the condition key {{key}}. We consider the key to
 be sensitive and wildcards are not allowed to be used with sensitive keys. Refer
 to https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-
keys.html for the list of sensitive keys.
```

```
Overly permissive use of wildcard with aws:userID. Use the wildcard on the right side
 of colon or as the only character on the left side of a colon.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Overly permissive use of wildcard for the {{key}} condition key. Use
 the wildcard preceded by 6 consecutive characters."
```

```
"findingDetails": "The wildcard is not allowed in the condition key {{key}}.
 We consider the key to be sensitive and wildcards are not allowed to be used
 with sensitive keys. Refer to https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_condition-keys.html for the list of sensitive keys."
```

```
"findingDetails": "Overly permissive use of wildcard with aws:userID. Use the wildcard
 on the right side of colon or as the only character on the left side of a colon."
```

Resolver o erro

Há três mensagens de descoberta possíveis para esse erro.

- Para a primeira mensagem de erro, torne o uso de curingas mais específico adicionando pelo menos seis caracteres consecutivos antes do curinga. Por exemplo, em vez de usar `*`, use `prefix-*` ou `prefix-*-suffix`. Isso reduz o escopo da condição e segue o princípio de privilégio mínimo.
- Para a segunda mensagem de erro, remova o curinga da chave de condição especificada. Curingas não são permitidas com chaves de condição confidenciais por motivos de segurança. Substitua o curinga por valores específicos que correspondam ao padrão de acesso pretendido ou considere usar outra chave de condição, não confidencial, que seja compatível com caracteres curingas.
- Para a primeira mensagem de erro, modifique o uso do curinga `aws:userID` para ser mais restritivo. Coloque o curinga do lado direito do sinal de dois-pontos (depois do ID da conta) ou use-a como o único caractere do lado esquerdo do sinal de dois-pontos. Por exemplo, use `AIDACKCEVSQ6C2EXAMPLE:*` ou `*:*` vez de `*`.

Termos relacionados

- Elementos da política do IAM: condição
- Condições com várias chaves ou valores de contexto
- AWS Chaves de contexto de condições globais da
- Identificadores do IAM

Erro: o DynamoDB retorna a capacidade consumida inválida

Código de problema: DYNAMODB_INVALID_RETURN_CONSUMED_CAPACITY

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB invalid return consumed capacity: The value for
 dynamodb:ReturnConsumedCapacity is not valid. Update {{value}} to either TOTAL or
 NONE.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The value for dynamodb:ReturnConsumedCapacity is not valid. Update
 {{value}} to either TOTAL or NONE."
```

Resolver o erro

Atualize o valor da condição para um dos dois valores que o DynamoDB fornece: `TOTAL` ou `NONE`. Os valores diferenciam maiúsculas de minúsculas em `StringEquals` e `StringLike`, portanto, uma variante minúscula, como `total`, não é válida. O operador `StringEqualsIgnoreCase` aceita qualquer capitalização. Uma condição que especifica um valor inválido nunca corresponde a nenhuma solicitação, portanto, a instrução não tem efeito.

Erro: valores de retorno inválidos do DynamoDB

Código de problema: DYNAMODB_INVALID_RETURN_VALUES

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB invalid return values: The value for dynamodb:ReturnValues is not valid.
 Update {{value}} to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The value for dynamodb:ReturnValues is not valid. Update {{value}}
 to one of NONE, ALL_OLD, UPDATED_OLD, ALL_NEW, or UPDATED_NEW."
```

Resolver o erro

Atualize o valor da condição para um dos cinco valores que o DynamoDB fornece: `NONE`, `ALL_OLD`, `UPDATED_OLD`, `ALL_NEW` ou `UPDATED_NEW`. Os valores diferenciam maiúsculas de minúsculas em `StringEquals` e `StringLike`, portanto, uma variante minúscula, como `updated_new`, não é válida. O operador `StringEqualsIgnoreCase` aceita qualquer capitalização. Uma condição que especifica um valor inválido nunca corresponde a nenhuma solicitação, portanto, a instrução não tem efeito.

Erro: valor de seleção inválido do DynamoDB

Código de problema: DYNAMODB_INVALID_SELECT_VALUE

Tipo de descoberta: ERROR

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB invalid select value: The value for dynamodb:Select is not valid. Update
 {{value}} to one of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or
 COUNT.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The value for dynamodb:Select is not valid. Update {{value}} to one
 of ALL_ATTRIBUTES, ALL_PROJECTED_ATTRIBUTES, SPECIFIC_ATTRIBUTES, or COUNT."
```

Resolver o erro

Atualize o valor da condição para um dos quatro valores que o DynamoDB fornece: `ALL_ATTRIBUTES`, `ALL_PROJECTED_ATTRIBUTES`, `SPECIFIC_ATTRIBUTES` ou `COUNT`. Os valores diferenciam maiúsculas de minúsculas em `StringEquals` e `StringLike`, portanto, uma variante minúscula, como `specific_attributes`, não é válida. O operador `StringEqualsIgnoreCase` aceita qualquer capitalização. Uma condição que especifica um valor inválido nunca corresponde a nenhuma solicitação, portanto, a instrução não tem efeito.

Aviso geral: Create SLR with NotResource (Crie uma SLR com NotResource)

Código do problema: CREATE_SLR_WITH_NOT_RESOURCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Resolução do aviso geral

A ção `iam:CreateServiceLinkedRole` concede permissão para criar uma função do IAM que permite que um produto da AWS execute ações em seu nome. O uso de `iam:CreateServiceLinkedRole` em uma política com o elemento `NotResource` pode permitir a criação de funções vinculadas ao serviço não intencionais para vários recursos. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`.

- Operação CreateServiceLinkedRole
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Resource

Aviso geral: Create SLR with star in action and NotResource (Crie uma SLR com uma estrela na ação e em NotResource)

Código do problema: CREATE_SLR_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
```

```
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Resolução do aviso geral

A ção `iam:CreateServiceLinkedRole` concede permissão para criar uma função do IAM que permite que um produto da AWS execute ações em seu nome. Políticas com um caractere curinga (*) em `Action` e que incluem o elemento `NotResource` podem permitir a criação de funções vinculadas ao serviço não intencionais para vários recursos. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`.

- Operação CreateServiceLinkedRole
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Resource

Aviso geral: Create SLR with NotAction and NotResource (Crie uma SLR com NotAction e NotResource)

Código do problema: CREATE_SLR_WITH_NOT_ACTION_AND_NOT_RESOURCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Resolução do aviso geral

A ção `iam:CreateServiceLinkedRole` concede permissão para criar uma função do IAM que permite que um produto da AWS execute ações em seu nome. O uso do elemento `NotAction` com o elemento `NotResource` pode permitir a criação de funções vinculadas ao serviço não intencionais para vários recursos. A AWS recomenda que você reescreva a política para permitir `iam:CreateServiceLinkedRole` em uma lista limitada de ARNs no elemento `Resource`. Você também pode adicionar `iam:CreateServiceLinkedRole` ao elemento `NotAction`.

- Operação CreateServiceLinkedRole
- Elementos de política JSON do IAM: NotAction
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Resource

Aviso geral: Create SLR with star in resource (Crie uma SLR com uma estrela no recurso)

Código do problema: CREATE_SLR_WITH_STAR_IN_RESOURCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Resolução do aviso geral

A ção `iam:CreateServiceLinkedRole` concede permissão para criar uma função do IAM que permite que um produto da AWS execute ações em seu nome. O uso de `iam:CreateServiceLinkedRole` em uma política com um caractere curinga (*) no elemento `Resource` pode permitir a criação de funções vinculadas ao serviço não intencionais para vários recursos. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`.

- Operação CreateServiceLinkedRole
- Elementos de política JSON do IAM: Resource

AWSPolíticas gerenciadas pela com este aviso geral

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

Alguns desses casos de uso são para usuários avançados dentro da conta. As seguintes políticas gerenciadas pela AWS fornecem acesso de usuário avançado e concedem permissões para criar funções vinculadas ao serviço para qualquer produto da AWS. A AWS recomenda que você anexe as seguintes políticas gerenciadas pela AWS somente para identidades do IAM que você considere usuários avançados.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy: esta política gerenciada pela AWS fornece permissões para uso pela função vinculada ao serviço do AWS Organizations. Essa função permite que o Organizations crie funções adicionais vinculadas ao serviço para outros serviços no AWS Organizations.

Aviso geral: Create SLR with star in action (Crie uma SLR com uma estrela na ação)

Código do problema: CREATE_SLR_WITH_STAR_IN_ACTION_AND_RESOURCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Resolução do aviso geral

A ção `iam:CreateServiceLinkedRole` concede permissão para criar uma função do IAM que permite que um produto da AWS execute ações em seu nome. Políticas com um caractere curinga (*) nos elementos `Action` e `Resource` podem permitir a criação de funções vinculadas ao serviço não intencionais para vários recursos. Isso permite a criação de uma função vinculada ao serviço quando você especifica `"Action": "*"`, `"Action": "iam:*"` ou `"Action": "iam:Create*"`. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`.

- Operação CreateServiceLinkedRole
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: Resource

AWSPolíticas gerenciadas pela com este aviso geral

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

Alguns desses casos de uso são para administradores de sua conta. As seguintes políticas gerenciadas pela AWS fornecem acesso de administrador e concedem permissões para criar funções vinculadas ao serviço para qualquer produto da AWS. A AWS recomenda que você anexe

as seguintes políticas gerenciadas pela AWS somente para identidades do IAM que você considere administradores.

- AdministratorAccess
- IAMFullAccess

Aviso geral: Create SLR with star in resource and NotAction (Crie uma SLR com uma estrela no frecurso e em NotAction)

Código do problema: CREATE_SLR_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Resolução do aviso geral

A ção `iam:CreateServiceLinkedRole` concede permissão para criar uma função do IAM que permite que um produto da AWS execute ações em seu nome. O uso do elemento `NotAction` em uma política com um caractere curinga (*) no elemento `Resource` pode permitir a criação de funções vinculadas ao serviço não intencionais para vários recursos. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`. Você também pode adicionar `iam:CreateServiceLinkedRole` ao elemento `NotAction`.

- Operação CreateServiceLinkedRole

- Elementos de política JSON do IAM: NotAction
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: Resource

Aviso geral: Deprecated global condition key (Chave de condição global defasada)

Código do problema: DEPRECATED_GLOBAL_CONDITION_KEY

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Resolução do aviso geral

A política inclui uma chave de condição global defasada. Atualize a chave de condição no par de chave-valor da condição para usar uma chave de condição global com suporte.

- Chaves de condições globais

Aviso geral: Invalid date value (Valor de data inválido)

Código do problema: INVALID_DATE_VALUE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Resolução do aviso geral

O tempo Epoch Unix descreve um ponto no tempo que decorre desde 1 de janeiro de 1970, menos os segundos intercalares. O tempo Epoch pode não resultar no tempo exato que você espera. A AWS recomenda que você use o padrão W3C para formatos de data e hora. Por exemplo, você pode especificar uma data completa, como AAAA-MM-DD (1997-07-16), ou também pode anexar o tempo ao segundo, como AAAA-MM-DDThh:mm:ssTZD (1997-07-16T19:20:30+01:00).

- Formatos de data e hora W3C
- Elementos de política JSON do IAM: Version
- Chave de condição global aws:CurrentTime

Aviso geral: Invalid role reference (Referência de função inválida)

Código do problema: INVALID_ROLE_REFERENCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Resolução do aviso geral

AWSA recomenda que você especifique o nome do recurso da Amazon (ARN) para uma função do IAM em vez do ID da entiddade. Quando o IAM salvar a política, ele transformará o ARN no ID de entidade da função existente. A AWS inclui uma precaução de segurança. Se alguém excluir e recriar a função, ela terá um novo ID e a política não corresponderá ao novo ID da função.

- Especificar uma entidade: funções do IAM
- ARNs do IAM
- IDs exclusivos do IAM

Aviso geral: Invalid user reference (Referência de usuário inválida)

Código do problema: INVALID_USER_REFERENCE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Resolução do aviso geral

AWSA recomenda que você especifique o nome do recurso da Amazon (ARN) para um usuário do IAM em vez do ID da entidade. Quando o IAM salvar a política, ele transformará o ARN no ID de entidade do usuário existente. A AWS inclui uma precaução de segurança. Se alguém excluir e recriar o usuário, ele terá um novo ID e a política não corresponderá ao novo ID do usuário.

- Especificar uma entidade: usuários do IAM

- ARNs do IAM
- IDs exclusivos do IAM

Aviso geral: Missing version (Versão ausente)

Código do problema: MISSING_VERSION

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Resolução do aviso geral

A AWS recomenda que você inclua o parâmetro `Version` opcional na sua política. Se você não incluir um elemento de versão, o valor padrão será `2012-10-17`, mas recursos mais recentes, como variáveis de política, não funcionarão com a sua política. Por exemplo, as variáveis como `${aws:username}` não serão reconhecidas como variáveis e serão tratadas como strings literais na política.

- Elementos de política JSON do IAM: Version

Aviso geral: Unique Sids recommended (É recomendável o uso de Sids exclusivos)

Código do problema: UNIQUE_SIDS_RECOMMENDED

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Resolução do aviso geral

AWSA recomenda que você use IDs de instrução exclusivos. O elemento `Sid` (ID de instrução) permite inserir um identificador opcional fornecido para a instrução de política. É possível atribuir um valor de ID de instrução a cada instrução em uma matriz de instruções usando o elemento `SID`.

Termos relacionados

- Elementos de política JSON do IAM: Sid

Aviso geral: Wildcard without like operator (Caractere curinga sem o operador Like)

Código do problema: WILDCARD_WITHOUT_LIKE_OPERATOR

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Resolução do aviso geral

A estrutura do elemento `Condition` requer que você use um operador de condição e um par de chave-valor. Quando você especifica um valor de condição que usa um caractere curinga (*,?), você deve usar a versão `Like` do operador de condição. Por exemplo, em vez do operador de condição de string `StringEquals`, use`StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- Elementos de política JSON do IAM: operadores de condição
- Elementos de política JSON do IAM: Condition

AWSPolíticas gerenciadas pela com este aviso geral

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

Os seguintes exemplos de políticas gerenciadas pela AWS incluem caracteres curingas no valor da condição sem um operador de condição que inclua `Like` para correspondência de padrões. Ao usar a política gerenciada pela AWS como referência para criar sua política gerenciada pelo cliente, a AWS recomenda que você use um operador de condição que suporte a correspondência de padrões com caracteres curingas (*,?), como `StringLike`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Aviso geral: Policy size exceeds identity policy quota (O tamanho da política excede a cota da política de identidade)

Código do problema: POLICY_SIZE_EXCEEDS_IDENTITY_POLICY_QUOTA

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
```

```
maximum for inline and managed policies. We recommend that you use multiple granular
policies.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Resolução do aviso geral

É possível anexar até 20 políticas gerenciadas a uma identidade do IAM (usuário, grupo de usuários ou perfil). No entanto, o tamanho de cada política gerenciada não pode exceder a cota padrão de 6.144 caracteres. O IAM não conta espaços em branco ao calcular o tamanho de uma política em relação a essa cota. As cotas, também conhecidas como limites na AWS, são os valores máximos para recursos, ações e itens na sua conta da AWS.

Além disso, você pode adicionar quantas políticas em linha desejar a uma identidade do IAM. Entretanto, o tamanho de todas as políticas em linha por identidade não pode exceder a cota especificada.

Se sua política for maior do que a cota, você poderá organizar sua política em várias instruções e agrupar as instruções em várias políticas.

Termos relacionados

- Cotas de caracteres do IAM e do AWS STS
- Várias instruções e várias políticas
- Políticas gerenciadas pelo cliente do IAM
- Visão geral das políticas de JSON
- Gramática de políticas de JSON do IAM

AWSPolíticas gerenciadas pela com este aviso geral

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

As seguintes políticas gerenciadas pela AWS concedem permissões para ações em vários produtos da AWS e excedem o tamanho máximo da política. Ao usar a política gerenciada pela AWS como uma referência para criar sua política gerenciada, você deve dividir a política em várias políticas.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Aviso geral: o tamanho da política excede a cota da política de recurso

Código do problema: POLICY_SIZE_EXCEEDS_RESOURCE_POLICY_QUOTA

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Resolução do aviso geral

Políticas baseadas em recursos são documentos de política JSON que você anexa a um recurso, como um bucket do Amazon S3. Essas políticas concedem permissão para a entidade principal especificada executar ações específicas nesse recurso e definem sob quais condições isso se aplica. O tamanho das políticas baseadas em recursos não pode exceder a cota determinada para o respectivo recurso. As cotas, também conhecidas como limites na AWS, são os valores máximos para recursos, ações e itens na sua conta da AWS.

Se sua política for maior do que a cota, você poderá organizar sua política em várias instruções e agrupar as instruções em várias políticas.

Termos relacionados

- Políticas baseadas em recursos
- Políticas de bucket do Amazon S3
- Várias instruções e várias políticas
- Visão geral das políticas de JSON
- Gramática de políticas de JSON do IAM

Aviso geral: Type mismatch (Incompatibilidade de tipo)

Código do problema: TYPE_MISMATCH

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Resolução do aviso geral

Atualize o texto para usar o tipo de dados do operador de condição com suporte.

Por exemplo, a chave de condição global `aws:MultiFactorAuthPresent` requer um operador de condição com o tipo de dados `Boolean`. Se você fornecer uma data ou um número inteiro, o tipo de dados não corresponderá.

Termos relacionados

- Chaves de condições globais

- Elementos de política JSON do IAM: operadores de condição

Aviso geral: Type mismatch Boolean (Incompatibilidade de tipo booliano)

Código do problema: TYPE_MISMATCH_BOOLEAN

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Resolução do aviso geral

Atualize o texto para usar um tipo de dados do operador de condição booliana, como `true` ou `false`.

Por exemplo, a chave de condição global `aws:MultiFactorAuthPresent` requer um operador de condição com o tipo de dados `Boolean`. Se você fornecer uma data ou um número inteiro, o tipo de dados não corresponderá.

Termos relacionados

- Operadores de condição booliana
- Elementos de política JSON do IAM: operadores de condição

Aviso geral: Type mismatch date (Incompatibilidade de tipo de data)

Código do problema: TYPE_MISMATCH_DATE

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Resolução do aviso geral

Atualize o texto para usar o tipo de dados do operador de condição de data, em um `YYYY-MM-DD` ou outro formato de data e hora ISO 8601.

Termos relacionados

- Operadores de condição de data
- Elementos de política JSON do IAM: operadores de condição

Aviso geral: Type mismatch number (Incompatibilidade de tipo de número)

Código do problema: TYPE_MISMATCH_NUMBER

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Resolução do aviso geral

Atualize o texto para usar o tipo de dados do operador de condição numérica.

Termos relacionados

- Operadores de condição numéricos
- Elementos de política JSON do IAM: operadores de condição

Aviso geral: Type mismatch string (Incompatibilidade de tipo de string)

Código do problema: TYPE_MISMATCH_STRING

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Resolução do aviso geral

Atualize o texto para usar o tipo de dados do operador de condição de string.

Termos relacionados

- Operadores de condição de strings
- Elementos de política JSON do IAM: operadores de condição

Aviso geral: Specific github repo and branch recommended (Repositório e ramificação específicos do github recomendado)

Código do problema: SPECIFIC_GITHUB_REPO_AND_BRANCH_RECOMMENDED

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name. If the subject claim does not include the branch, specify
 it in a token.actions.githubusercontent.com:ref condition key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value
 of token.actions.githubusercontent.com:sub with the repository and branch
 name. If the subject claim does not include the branch, specify it in a
 token.actions.githubusercontent.com:ref condition key."
```

Resolução do aviso geral

Se você usar o GitHub como um IdP OIDC, a prática recomendada é limitar as entidades que podem assumir a função associado ao IdP do IAM. Quando inclui uma instrução de `Condition` na política de confiança da função, você pode limitar a função a uma organização, repositório ou ramificação específica do GitHub. É possível usar a chave de condição `token.actions.githubusercontent.com:sub` para limitar o acesso. Recomendamos que você limite a condição a um conjunto específico de repositórios ou ramificações. Se você usar um curinga (`*`) no `token.actions.githubusercontent.com:sub`, ações do GitHub de organizações ou repositórios fora do seu controle poderão assumir funções associadas ao IdP do IAM do GitHub na sua conta da AWS.

Especifique o repositório em `token.actions.githubusercontent.com:sub`. Você pode especificar a ramificação no mesmo valor ou em uma chave de condição

`token.actions.githubusercontent.com:ref` separada. Quando a declaração do assunto não incluir a ramificação — por exemplo, quando ela nomeia um ambiente GitHub — especifique a ramificação em `token.actions.githubusercontent.com:ref`.

Termos relacionados

- Configurar uma função para o provedor de identidades do OIDC do GitHub

Aviso geral: Policy size exceeds role trust policy quota (Tamanho da política excede a cota da política de confiança da função)

Código do problema: POLICY_SIZE_EXCEEDS_ROLE_TRUST_POLICY_QUOTA

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Resolução do aviso geral

O IAM e o AWS STS têm cotas que limitam o tamanho das políticas de confiança das funções. Os caracteres na política de confiança da função, excluindo os espaços em branco, excedem o máximo de caracteres. Recomendamos que você solicite um aumento da cota de tamanho da política de confiança da função usando as Service Quotas e o AWS Support Center Console.

Termos relacionados

- Cotas, requisitos de nome e limites de caracteres do IAM e do AWS STS

Aviso geral: falta a chave de condição de entidade principal relacionada na RCP

Código do problema: RCP_MISSING_RELATED_PRINCIPAL_CONDITION_KEY

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP missing related principal condition key: RCPs impact IAM roles, users, and AWS
 service principals. To prevent unintended impact to services acting on your behalf
 using a service principal, an additional statement should be added to the Condition
 block "BoolIfExists": { "aws:PrincipalIsAWSService": "false"} whenever a principal key
 {{conditionKeyName}} is used.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to services acting on your behalf using a service principal,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "false"} whenever a principal key {{conditionKeyName}}
 is used."
```

Resolução do aviso geral

As políticas de controle de recursos (RCPs) do AWS Organizations podem afetar perfis do IAM, usuários e entidades principais do AWS service (Serviço da AWS). Para evitar um impacto indesejado nos serviços que atuam em seu nome usando uma entidade principal de serviço, adicione a instrução a seguir ao seu elemento `Condition`:

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "false"}
```

Termos relacionados

- Sintaxe de RCP
- Propriedades da entidade principal

Aviso geral: falta a chave de condição de entidade principal de serviço relacionado no RCP

Código do problema: RCP_MISSING_RELATED_SERVICE_PRINCIPAL_CONDITION_KEY

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP missing related service principal condition key: RCPs impact IAM roles, users,
 and AWS service principals. To prevent unintended impact to your principals,
 an additional statement should be added to the Condition block "BoolIfExists":
 { "aws:PrincipalIsAWSService": "true"} whenever the key {{conditionKeyName}} is used.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "RCPs impact IAM roles, users, and AWS service principals. To prevent
 unintended impact to your principals, an additional statement should be added to the
 Condition block "BoolIfExists": { "aws:PrincipalIsAWSService": "true"} whenever the
 key {{conditionKeyName}} is used."
```

Resolução do aviso geral

As políticas de controle de recursos (RCPs) do AWS Organizations podem afetar perfis do IAM, usuários e entidades principais do AWS service (Serviço da AWS). Para evitar um impacto indesejado em suas entidades principais, adicione a instrução a seguir ao seu elemento `Condition`:

```
"BoolIfExists": { "aws:PrincipalIsAWSService": "true"}
```

Termos relacionados

- Sintaxe de RCP
- Propriedades da entidade principal

Aviso geral: RCP sem verificação de nulidade da chave de condição do serviço

Código do problema: RCP_MISSING_SERVICE_CONDITION_KEY_NULL_CHECK

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
RCP missing service condition key null check: The specified service may have a service
 integration that does not require the use of the {{conditionKeyName}} condition
 key. To prevent unintended impact to services acting on your behalf using a service
 principal, an additional statement should be added to the Condition block "Null":
 { "aws:SourceAccount": "false"} or "Null": { "aws:SourceArn": "false"} whenever the
 key {{conditionKeyName}} is used.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The specified service may have a service integration that does
 not require the use of the {{conditionKeyName}} condition key. To prevent unintended
 impact to services acting on your behalf using a service principal, an additional
 statement should be added to the Condition block "Null": { "aws:SourceAccount":
 "false"} or "Null": { "aws:SourceArn": "false"} whenever the key {{conditionKeyName}}
 is used."
```

Resolução do aviso geral

As políticas de controle de recursos (RCPs) do AWS Organizations podem afetar perfis do IAM, usuários e entidades principais do AWS service (Serviço da AWS). Para evitar um impacto indesejado nos serviços que atuam em seu nome usando uma entidade principal de serviço, adicione uma das instruções a seguir ao elemento `Condition` sempre que a chave especificada for usada:

```
"Null": { "aws:SourceAccount": "false"}
```

ou

```
"Null": { "aws:SourceArn": "false"}
```

Termos relacionados

- Sintaxe de RCP

- Propriedades da entidade principal

Aviso geral: use a chave de condição somente com serviços compatíveis

Código de problema: USE_CONDITION_KEY_ONLY_WITH_SUPPORTED_SERVICES

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Use condition key only with supported services: The condition key {{key}} works only
 with specific AWS services and must be scoped to supported services in your policies.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{key}} works only with specific AWS services and
 must be scoped to supported services in your policies.
```

Resolução do aviso geral

Revise a documentação da AWS para identificar quais Serviços da AWS suportam essa chave de condição. Se algum serviço em sua política não oferecer suporte à chave de condição, modifique sua política para definir o escopo da chave de condição somente para os serviços da Serviços da AWS que a suportam.

Termos relacionados

- `aws:VpceAccount`
- `aws:VpceOrgID`
- `aws:VpceOrgPaths`
- Ações, recursos e chaves de condição para o Serviços da AWS

Aviso geral: operação de substituição do DynamoDB por atributos

Código de problema: DYNAMODB_REPLACE_OPERATION_WITH_ATTRIBUTES

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB replace operation with attributes: The {{action}} action replaces the entire
 item, so dynamodb:Attributes cannot restrict access at the attribute level. That
 condition only applies to UpdateItem; use dynamodb:UpdateItem for attribute-level
 access control.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{action}} action replaces the entire item, so
 dynamodb:Attributes cannot restrict access at the attribute level. That condition only
 applies to UpdateItem; use dynamodb:UpdateItem for attribute-level access control."
```

Resolução do aviso geral

A condição `dynamodb:Attributes` não limita `dynamodb:PutItem` ou `dynamodb:BatchWriteItem`. Estas ações substituem o item inteiro, então um chamador pode eliminar um atributo restrito simplesmente deixando-o fora do item que ele gravou. Para controlar quais atributos um chamador pode alterar, conceda `dynamodb:UpdateItem` e mantenha a condição `dynamodb:Attributes` nessa instrução. Se você não pretendia controlar o nível do atributo aqui, remova a condição `dynamodb:Attributes`.

Aviso geral: a verificação do DynamoDB ignora as chaves principais

Código de problema: DYNAMODB_SCAN_BYPASSES_LEADING_KEYS

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB scan bypasses leading keys: The dynamodb:Scan operation returns all items
 in the table, bypassing the dynamodb:LeadingKeys restriction. Consider removing
 dynamodb:Scan.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The dynamodb:Scan operation returns all items in the table,
 bypassing the dynamodb:LeadingKeys restriction. Consider removing dynamodb:Scan."
```

Resolução do aviso geral

A condição `dynamodb:Scan` lê a tabela inteira, portanto, a condição `dynamodb:LeadingKeys` nesta instrução não limita quais itens um chamador pode ler. Remova a condição `dynamodb:Scan` ou a conceda em outra instrução que não dependa da condição `dynamodb:LeadingKeys`.

Aviso geral: condição errada para ação do DynamoDB

Código de problema: DYNAMODB_WRONG_CONDITION_FOR_ACTION

Tipo de descoberta: GENERAL_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB wrong condition for action: The condition key {{conditionKey}} does
 not apply to the {{action}} action. Use dynamodb:Select with read actions and
 dynamodb:ReturnValues with write actions.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{conditionKey}} does not apply to the {{action}}
 action. Use dynamodb:Select with read actions and dynamodb:ReturnValues with write
 actions."
```

Resolução do aviso geral

Esta instrução usa a condição `dynamodb:Select` com uma ação de gravação ou a condição `dynamodb:ReturnValues` com uma ação de leitura. O DynamoDB fornece a condição `dynamodb:Select` somente para ações de leitura e a condição `dynamodb:ReturnValues` somente para ações de gravação, portanto, a condição aqui nunca entra em vigor e não protege nada. Associe cada chave com uma ação de suporte ou remova-a. Esta verificação se aplica às instruções `Allow` e `Deny`.

Aviso de segurança: chave de condição não confiável

Código do problema: UNTRUSTWORTHY_CONDITION_KEY

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Untrustworthy condition key: The {{key}} condition key is not recommended for access
 control as it can be spoofed/manipulated by the caller.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{key}} condition key is not recommended for access control as
 it can be spoofed/manipulated by the caller."
```

Resolução do aviso de segurança

Não use essa chave de condição para controlar acesso. O chamador pode potencialmente manipular ou falsificar o valor da chave, o que cria um risco de segurança.

Termos relacionados

- AWS Chaves contextuais de condições globais da

Aviso de segurança: Allow with NotPrincipal (Permitir com NotPrincipal)

Código do problema: ALLOW_WITH_NOT_PRINCIPAL

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Resolução do aviso de segurança

O uso de `"Effect": "Allow"` com `NotPrincipal` pode ser excessivamente permissivo. Por exemplo, isso pode conceder permissões a entidades principais anônimas. A AWS recomenda que você especifique entidades principais que precisem de acesso com o elemento `Principal`. Como alternativa, você pode permitir acesso amplo e, em seguida, adicionar outra instrução que use o elemento `NotPrincipal` com `“Effect”: “Deny”`.

- AWS Elementos de política JSON da : entidade
- AWS Elementos de política JSON da : NotPrincipal

Aviso de segurança: ForAllValues with single valued key (ForAllValues com chave de valor único)

Código do problema: FORALLVALUES_WITH_SINGLE_VALUED_KEY

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Resolução do aviso de segurança

A AWS recomenda o uso de `ForAllValues` apenas com condições de vários valores. O operador de conjunto `ForAllValues` testa se o valor de cada membro do conjunto de solicitações é um subconjunto do conjunto de chaves de condição. A condição retornará "verdadeiro" se cada valor de chave na solicitação corresponder a pelo menos um valor na política. Ela também retornará "verdadeiro" se não houver chaves na solicitação, ou se os valores de chave forem resolvidos para um conjunto de dados nulo, como uma string vazia.

Para saber se uma condição suporta um único valor ou vários valores, consulte a página Ações, recursos e chaves de condição para o serviço. Chaves de condição com o prefixo de tipo de dados `ArrayOf` são chaves de condição com vários valores. Por exemplo, o Amazon SES suporta chaves com valores únicos (`String`) e o tipo de dados com vários valores `ArrayOfString`.

- Chaves de contexto de múltiplos valores

Aviso de segurança: Pass role with NotResource (Passar função com NotResource)

Código do problema: PASS_ROLE_WITH_NOT_RESOURCE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Resolução do aviso de segurança

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Para permitir isso, você deve conceder a permissão `iam:PassRole` a uma identidade (usuário, grupo de usuários ou função). O uso de `iam:PassRole` em uma política com o elemento

`NotResource` pode permitir que suas entidades principais acessem mais serviços ou recursos do que o pretendido. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`. Além disso, você pode reduzir permissões para um único serviço usando a chave de condição `iam:PassedToService`.

- Passar uma função para um serviço
- iam:PassedToService
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Resource

Aviso de segurança: Pass role with star in action and NotResource (Passar a função com uma estrela na ação e NotResource)

Código do problema: PASS_ROLE_WITH_STAR_IN_ACTION_AND_NOT_RESOURCE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Resolução do aviso de segurança

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Para permitir isso, você deve conceder a permissão `iam:PassRole` a uma identidade (usuário, grupo de usuários ou função). Políticas com um caractere curinga (*) em `Action` e que incluem o elemento `NotResource` pode permitir que suas entidades principais acessem mais serviços

ou recursos do que o pretendido. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`. Além disso, você pode reduzir permissões para um único serviço usando a chave de condição `iam:PassedToService`.

- Passar uma função para um serviço
- iam:PassedToService
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Resource

Aviso de segurança: Pass role with NotAction and NotResource (Passe a função com NotAction e NotResource)

Código do problema: PASS_ROLE_WITH_NOT_ACTION_AND_NOT_RESOURCE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Resolução do aviso de segurança

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Para permitir isso, você deve conceder a permissão `iam:PassRole` a uma identidade (usuário, grupo de usuários ou função). O uso do elemento `NotAction` e a listagem de alguns recursos no elemento `NotResource` pode permitir que suas entidades principais acessem mais serviços ou recursos do que o pretendido. A AWS recomenda que você especifique ARNs permitidos no

elemento `Resource`. Além disso, você pode reduzir permissões para um único serviço usando a chave de condição `iam:PassedToService`.

- Passar uma função para um serviço
- iam:PassedToService
- Elementos de política JSON do IAM: NotAction
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Resource

Aviso de segurança: Pass role with star in resource (Passar função com uma estrela no recurso)

Código do problema: PASS_ROLE_WITH_STAR_IN_RESOURCE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Resolução do aviso de segurança

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Para permitir isso, você deve conceder a permissão `iam:PassRole` a uma identidade (usuário,

grupo de usuários ou função). Políticas que permitem `iam:PassRole` e que incluem um caractere curinga (*) no elemento `Resource` podem permitir que suas entidades principais acessem mais serviços ou recursos do que o pretendido. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`. Além disso, você pode reduzir permissões para um único serviço usando a chave de condição `iam:PassedToService`.

Alguns produtos da AWS incluem seus respectivos namespaces de serviço no nome da função. Essa verificação de política leva essas convenções em consideração ao analisar a política para gerar descobertas. Por exemplo, o ARN de recurso a seguir pode não gerar uma descoberta:

```
arn:aws:iam::*:role/Service*
```

- Passar uma função para um serviço
- iam:PassedToService
- Elementos de política JSON do IAM: Resource

AWSPolíticas gerenciadas pela com este aviso de segurança

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

Um desses casos de uso destina-se a administradores dentro da sua conta. As políticas gerenciadas pela AWS a seguir fornecem acesso de administrador e concedem permissões para passar qualquer função do IAM para qualquer serviço. A AWS recomenda que você anexe as seguintes políticas gerenciadas pela AWS apenas às identidades do IAM que você considera administradores.

- AdministratorAccess-Amplify

As políticas gerenciadas pela AWS incluem permissões para `iam:PassRole` com um caractere curinga (*) no recurso e estão tornando-se defasadas. Para cada uma dessas políticas, atualizamos as diretrizes de permissão, como recomendar uma nova política gerenciada pela AWS que ofereça suporte ao caso de uso. Para visualizar alternativas a essas políticas, consulte os guias de cada serviço.

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess

- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOPsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonEC2ContainerServiceFullAccess

As políticas gerenciadas pela AWS a seguir fornecem permissões somente para funções vinculadas ao serviço, o que permite que os produtos da AWS realizem ações em seu nome. Você não pode anexar essas políticas às suas identidades do IAM.

- AWSServiceRoleForAmazonEKSNodegroup

Aviso de segurança: Pass role with star in action and resource (Passar função com uma estrela na ação e no recurso)

Código do problema: PASS_ROLE_WITH_STAR_IN_ACTION_AND_RESOURCE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
```

```
that you specify resource ARNs or add the iam:PassedToService condition key to your
statement."
```

Resolução do aviso de segurança

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Para permitir isso, você deve conceder a permissão `iam:PassRole` a uma identidade (usuário, grupo de usuários ou função). Políticas com um caractere curinga (*) nos elementos `Action` e `Resource`podem permitir que suas entidades principais acessem mais serviços ou recursos do que o pretendido. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`. Além disso, você pode reduzir permissões para um único serviço usando a chave de condição `iam:PassedToService`.

- Passar uma função para um serviço
- iam:PassedToService
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: Resource

AWSPolíticas gerenciadas pela com este aviso de segurança

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

Alguns desses casos de uso são para administradores de sua conta. As políticas gerenciadas pela AWS a seguir fornecem acesso de administrador e concedem permissões para passar qualquer função do IAM para qualquer produto da AWS. A AWS recomenda que você anexe as seguintes políticas gerenciadas pela AWS apenas às identidades do IAM que você considera administradores.

- AdministratorAccess
- IAMFullAccess

Aviso de segurança: Pass role with star in resource and NotAction (Passar a função com uma estrela no recurso e NotAction)

Código do problema: PASS_ROLE_WITH_STAR_IN_RESOURCE_AND_NOT_ACTION

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Resolução do aviso de segurança

Para configurar muitos produtos da AWS é necessário passar uma função do IAM para o serviço. Para permitir isso, você deve conceder a permissão `iam:PassRole` a uma identidade (usuário, grupo de usuários ou função). O uso do elemento `NotAction` em uma política com um caractere curinga (*) no elemento `Resource` pode permitir que suas entidades principais acessem mais serviços ou recursos do que o pretendido. A AWS recomenda que você especifique ARNs permitidos no elemento `Resource`. Além disso, você pode reduzir permissões para um único serviço usando a chave de condição `iam:PassedToService`.

- Passar uma função para um serviço
- iam:PassedToService
- Elementos de política JSON do IAM: NotAction
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: Resource

Aviso de segurança: Missing paired condition keys (Chaves de condição emparelhadas ausentes)

Código do problema: MISSING_PAIRED_CONDITION_KEYS

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Resolução do aviso de segurança

Algumas chaves de condição são mais seguras quando emparelhadas com outras chaves de condição relacionadas. A AWS recomenda que você inclua as chaves de condição relacionadas no mesmo bloco de condições que a chave de condição existente. Isso torna as permissões concedidas por meio da política mais seguras.

Por exemplo, você pode usar a chave de condição `aws:VpcSourceIp` para comparar o endereço IP do qual uma solicitação foi feita com o endereço IP especificado na política. A AWS recomenda que você adicione a chave de condição `aws:SourceVPC` relacionada. Isso verifica se a solicitação é proveniente da VPC especificada na políticae do endereço IP que você especifica.

Termos relacionados

- `aws:VpcSourceIp` Chave de condição global da
- `aws:SourceVPC` Chave de condição global da
- Chaves de condições globais
- Elemento de condição
- Visão geral das políticas de JSON

Aviso de segurança: Deny with unsupported tag condition key for service (Negar com chave de condição de etiqueta não suportada para o serviço)

Código do problema: DENY_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Resolução do aviso de segurança

O uso de chaves de condição de etiqueta não suportadas no elemento `Condition` de uma política com `"Effect": "Deny"` pode ser excessivamente permissivo, porque a condição é ignorada para esse serviço. A AWS recomenda que você remova as ações de serviço que não oferecem suporte à chave de condição e crie outra instrução para negar acesso a recursos específicos para essas ações.

Se você usar a chave de condição `aws:ResourceTag` e uma ação de serviço não oferecer suporte a ela, a chave não será incluída no contexto da solicitação. Neste caso, a condição na instrução `Deny` sempre retorna `false` e a ação nunca é negada. Isso acontece mesmo se o recurso estiver etiquetado corretamente.

Quando um serviço oferece suporte à chave de condição `aws:ResourceTag`, você pode usar etiquetas para controlar o acesso aos recursos desse serviço. Isso é conhecido como controle de

acesso baseado em atributo (ABAC). Os serviços que não oferecem suporte a essas chaves exigem que você controle o acesso a recursos usando o controle de acesso baseado em recursos (RBAC).

> **Note**
>
> Alguns serviços permitem oferecer suporte à chave de condição `aws:ResourceTag` para um subconjunto de seus recursos e ações. O IAM Access Analyzer retorna descobertas para as ações de serviço incompatíveis. Por exemplo, o Amazon S3 é compatível com `aws:ResourceTag` para um subconjunto de seus recursos. Para visualizar todos os tipos de recursos disponíveis no Amazon S3 que oferecem suporte à chave de condição `aws:ResourceTag`, consulte Tipos de recursos definidos pelo Amazon S3 na Referência de autorização do serviço.

Por exemplo, suponhamos que você deseja negar acesso para desetiquetar recursos específicos de exclusão etiquetados com o par de chave-valor `status=Confidential`. Suponha também que o AWS Lambda permita etiquetar e desetiquetar recursos, mas não ofereça suporte à chave de condição `aws:ResourceTag`. Para negar as ações de exclusão para AWS App Mesh e AWS Backup se essa etiqueta estiver presente, use a chave de condição `aws:ResourceTag`. Para o Lambda, use uma convenção de nomenclatura de recurso que inclua o prefixo `"Confidential"`. Em seguida, inclua uma instrução separada que impeça a exclusão de recursos com essa convenção de nomenclatura.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
```

```
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> Não use a versão ...IfExists do operador de condição como uma solução alternativa para essa descoberta. Isso significa “Negue a ação se a chave estiver presente no contexto da solicitação e os valores forem correspondentes. Caso contrário, negue a ação.” No exemplo anterior, a inclusão da ação `lambda:DeleteFunction` na instrução `DenyDeleteSupported` com o operador `StringEqualsIfExists` sempre nega a ação. Para essa ação, a chave não está presente no contexto e cada tentativa de excluir esse tipo de recurso é negada, independentemente de o recurso estar etiquetado.

Termos relacionados

- Chaves de condições globais
- Comparação de ABAC com RBAC
- Elementos de política JSON do IAM: operadores de condição
- Elemento de condição
- Visão geral das políticas de JSON

Aviso de segurança: Deny NotAction with unsupported tag condition key for service (Negar NotAction com chave de condição de etiqueta não suportada para o serviço)

Código do problema:
DENY_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Resolução do aviso de segurança

O uso de chaves de condição de etiqueta no elemento `Condition` de uma política com o elemento `NotAction` e `"Effect": "Deny"` pode ser excessivamente permissivo. A condição é ignorada para ações de serviço que não suportam a chave de condição. A AWS recomenda que você reescreva a lógica para negar uma lista de ações.

Se você usar a chave de condição `aws:ResourceTag` com `NotAction`, quaisquer ações de serviço novas ou existentes que não suportem a chave não serão negadas. A AWS recomenda que você liste explicitamente as ações que você deseja negar. O IAM Access Analyzer retorna uma descoberta separada para ações listadas que não oferecem suporte para a chave de condição `aws:ResourceTag`. Para obter mais informações, consulte Aviso de segurança: Deny with unsupported tag condition key for service (Negar com chave de condição de etiqueta não suportada para o serviço).

Quando um serviço oferece suporte à chave de condição `aws:ResourceTag`, você pode usar etiquetas para controlar o acesso aos recursos desse serviço. Isso é conhecido como controle de acesso baseado em atributo (ABAC). Os serviços que não oferecem suporte a essas chaves exigem que você controle o acesso a recursos usando o controle de acesso baseado em recursos (RBAC).

Termos relacionados

- Chaves de condições globais
- Comparação de ABAC com RBAC

- Elementos de política JSON do IAM: operadores de condição
- Elemento de condição
- Visão geral das políticas de JSON

Aviso de segurança: restringir acesso à entidade principal de serviço

Código do problema: RESTRICT_ACCESS_TO_SERVICE_PRINCIPAL

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Resolução do aviso de segurança

É possível especificar Serviços da AWS no elemento `Principal` de uma política baseada em recursos usando uma entidade principal de serviço, que é um identificador desse serviço. Ao conceder acesso à entidade principal de serviço para atuar em seu nome, restrinja o acesso. É possível evitar políticas excessivamente permissivas usando as chaves de condição `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` para restringir o acesso a uma determinada fonte, como o ARN específico de um recurso, a Conta da AWS, o ID da organização ou os caminhos da organização. Restringir o acesso ajuda a evitar um problema de segurança chamado problema do substituto confuso.

Termos relacionados

- AWS service (Serviço da AWS) Entidades principais do

- AWS Chaves globais de condição da : aws:SourceAccount
- AWS Chaves globais de condição da : aws:SourceArn
- AWS Chaves de condições globais da : aws:SourceOrgId
- AWS Chaves de condições globais da : aws:SourceOrgPaths
- O problema "confused deputy

Aviso de segurança: Missing condition keys for oidc principal (Chaves de condição ausentes para entidade principal do oidc)

Código do problema: MISSING_CONDITION_KEY_FOR_OIDC_PRINCIPAL

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Resolução do aviso de segurança

Usar uma entidade principal do Open ID Connect sem uma condição pode ser excessivamente permissivo. Adicione chaves de condição com um prefixo que corresponda às entidades principais federadas do OIDC para garantir que somente o provedor de identidades pretendido assuma a função.

Termos relacionados

- Criar uma função para identidade da Web ou federação do OpenID Connect (console)

Aviso de segurança: Missing github repo condition key (Chave de condição do repositório Github ausente)

Código do problema: MISSING_GITHUB_REPO_CONDITION_KEY

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
 Add the token.actions.githubusercontent.com:sub condition key and specify the branch
 and repository name in the value.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

Resolução do aviso de segurança

Se você usar o GitHub como um IdP OIDC, a prática recomendada é limitar as entidades que podem assumir a função associado ao IdP do IAM. Quando inclui uma instrução de `Condition` na política de confiança da função, você pode limitar a função a uma organização, repositório ou ramificação específica do GitHub. É possível usar a chave de condição `token.actions.githubusercontent.com:sub` para limitar o acesso. Recomendamos que você limite a condição a um conjunto específico de repositórios ou ramificações. Se você não incluir essa condição, ações do GitHub de organizações ou repositórios fora do seu controle poderão assumir funções associadas ao IdP do IAM do GitHub na sua conta da AWS.

Termos relacionados

- Configurar uma função para o provedor de identidades do OIDC do GitHub

Aviso de segurança: operador semelhante a uma string com chaves de condição de ARN

Código do problema: STRING_LIKE_OPERATOR_WITH_ARN_CONDITION_KEYS

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
String like operator with ARN condition keys: Use the operator type {{allowed}} instead
 of operator {{operator}} for the condition key {{key}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Resolução do aviso de segurança

AWSA recomenda usar operadores de ARN em vez de operadores string ao comparar ARNs para garantir a restrição de acesso adequada com base nos valores de condição de ARN. Atualize o operador `StringLike` para o operador `ArnLike` em seu elemento `Condition` sempre que a chave especificada for usada.

Essas políticas gerenciadas pela AWS são exceções a esse aviso de segurança:

- AmazonSecurityLakeAdministrator
- AWSCodePipeline_FullAccess
- AWSCodePipeline_ReadOnlyAccess
- S3UnlockBucketPolicy
- SQSUnlockQueuePolicy

Termos relacionados

- Operadores de condição de nome do recurso da Amazon (ARN

- Operadores de condição de strings
- AWS Políticas gerenciadas pela

Aviso de segurança — atributos do DynamoDB sem seleção

Código do problema: DYNAMODB_ATTRIBUTES_WITHOUT_SELECT

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB attributes without select: Restricting dynamodb:Attributes without also
 setting dynamodb:Select to SPECIFIC_ATTRIBUTES allows all attributes to be returned
 on read requests that omit a projection expression. We recommend that you also set
 dynamodb:Select to SPECIFIC_ATTRIBUTES.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Restricting dynamodb:Attributes without also setting dynamodb:Select
 to SPECIFIC_ATTRIBUTES allows all attributes to be returned on read requests that
 omit a projection expression. We recommend that you also set dynamodb:Select to
 SPECIFIC_ATTRIBUTES."
```

Resolução do aviso de segurança

Adicione uma condição `StringEquals` que define `dynamodb:Select` como `SPECIFIC_ATTRIBUTES` na mesma instrução da condição `dynamodb:Attributes`. O DynamoDB avalia `dynamodb:Attributes` somente quanto às solicitações que especificam os atributos a serem retornados. Sem essa condição, uma solicitação de leitura que omite uma expressão de projeção retorna o item inteiro.

Termos relacionados

- Uso de condições de política do IAM para controle de acesso refinado
- Elementos de política JSON do IAM: operadores de condição

Aviso de segurança — prefixo S3 no contexto negativo

Código do problema: S3_PREFIX_IN_NEGATIVE_CONTEXT

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
S3 prefix in negative context: Using the s3:prefix condition key in a Deny statement or
 with a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the s3:prefix condition key in a Deny statement or with
 a negated string operator does not block listing the objects under the matching
 prefixes. A request that specifies a shorter prefix does not match the condition and
 can still list those objects. To restrict listing to specific prefixes, use s3:prefix
 with StringLike in an Allow statement, or use a Deny statement with StringNotLike."
```

Resolução do aviso de segurança

Para restringir a listagem a prefixos específicos, use um dos seguintes padrões:

- Permitir com StringLike — Conceda `s3:ListBucket` com uma condição `StringLike` em `s3:prefix` listando os prefixos que você deseja permitir. Solicitações que fornecem um prefixo diferente não são cobertas pela Permissão e não podem listar esses objetos.
- Negar com StringNotLike — Negue `s3:ListBucket` com uma condição `StringNotLike` em `s3:prefix` listando os prefixos que você deseja permitir. Uma solicitação que omite o prefixo ou fornece um prefixo mais curto não corresponde ao conjunto permitido e é negada.

Um `Deny` com `StringLike` ou um `Allow` com `StringNotLike` não restringe efetivamente o acesso porque um chamador pode ignorar a condição especificando um prefixo mais curto (por exemplo, `priv` em vez de `private/`) que não corresponde ao valor da condição.

Termos relacionados

- Exemplos de chave de condição do Amazon S3
- Elementos de política JSON do IAM: operadores de condição
- Elementos da política do IAM: condição

Aviso de segurança: ForAnyValue com tipo de reivindicação do público

Código do problema: FORANYVALUE_WITH_AUDIENCE_CLAIM_TYPE

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
ForAnyValue with audience claim type: Using ForAnyValue qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAnyValue:.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using ForAnyValue qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAnyValue:."
```

Resolução do aviso de segurança

A AWS recomenda que você não use operadores de conjunto `ForAnyValue` com chaves de condição com valor único. Use operadores de conjunto somente com chaves de condições de vários valores. Remova o operador de conjunto `ForAnyValue`.

Termos relacionados

- Chaves de contexto com valor único comparadas com chaves de contexto com vários valores
- Exemplos de políticas de chave de contexto com valor único

Aviso de segurança: atributos do DynamoDB sem valores de retorno

Código do problema: DYNAMODB_ATTRIBUTES_WITHOUT_RETURN_VALUES

Tipo de descoberta: SECURITY_WARNING

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
DynamoDB attributes without return values: Restricting dynamodb:Attributes without
 limiting dynamodb:ReturnValues allows write operations to return the entire item,
 which can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Restricting dynamodb:Attributes without limiting
 dynamodb:ReturnValues allows write operations to return the entire item, which
 can expose attributes outside the allowed list. We recommend that you restrict
 dynamodb:ReturnValues to NONE, UPDATED_OLD, or UPDATED_NEW."
```

Resolução do aviso de segurança

Adicione uma condição que limita `dynamodb:ReturnValues` a `NONE`, `UPDATED_OLD` ou `UPDATED_NEW` na mesma instrução como a condição `dynamodb:Attributes`. Sem esse limite, uma gravação que define `dynamodb:ReturnValues` como `ALL_OLD` ou `ALL_NEW` retorna o item inteiro, incluindo atributos fora dos permitidos.

Sugestão: Empty array action (Ação de matriz vazia)

Código do problema: EMPTY_ARRAY_ACTION

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Resolução da sugestão

As instruções devem incluir um elemento `Action` ou `NotAction` que inclua um conjunto de ações. Quando o elemento está vazio, a instrução de política não fornece permissões. Especifique ações no elemento `Action`.

- Elementos de política JSON do IAM: Action

Sugestão: Empty array condition (Condição de matriz vazia)

Código do problema: EMPTY_ARRAY_CONDITION

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Resolução da sugestão

A estrutura do elemento `Condition` opcional requer que você use um operador de condição e um par de chave-valor. Quando o valor da condição está vazio, a condição retorna `true` e a instrução de política não fornece permissões. Especifique um valor de condição.

- Elementos de política JSON do IAM: Condition

Sugestão: Empty array condition ForAllValues (Condição de matriz vazia ForAllValues)

Código do problema: EMPTY_ARRAY_CONDITION_FORALLVALUES

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Resolução da sugestão

A estrutura do elemento `Condition` requer que você use um operador de condição e um par de chave-valor. O operador de conjunto `ForAllValues` testa se o valor de cada membro do conjunto de solicitações é um subconjunto do conjunto de chaves de condição.

Quando você usa `ForAllValues` com uma chave de condição vazia, a condição corresponde somente se não houver chaves na solicitação. A AWS recomenda que, se você quiser testar se um contexto de solicitação está vazio, use o operador de condição `Null`.

- Chaves de contexto de múltiplos valores
- Operador de condição Null
- Elementos de política JSON do IAM: Condition

Sugestão: Empty array condition ForAnyValue (Condição de matriz vazia ForAnyValue)

Código do problema: EMPTY_ARRAY_CONDITION_FORANYVALUE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Resolução da sugestão

A estrutura do elemento `Condition` requer que você use um operador de condição e um par de chave-valor. O operador de conjunto `ForAnyValues` testa se pelo menos um membro do conjunto de valores de solicitação corresponde a pelo menos um membro do conjunto de valores de chave de condição.

Quando você usa `ForAnyValues` com uma chave de condição vazia, a condição nunca corresponde. Isso significa que a instrução não tem efeito sobre a política. A AWS recomenda que você reescreva a condição.

- Chaves de contexto de múltiplos valores
- Elementos de política JSON do IAM: Condition

Sugestão: Empty array condition IfExists (Condição de matriz vazia IfExists)

Código do problema: EMPTY_ARRAY_CONDITION_IFEXISTS

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Resolução da sugestão

O sufixo `...IfExists` edita um operador de condição. Isso significa que, se a chave da política estiver presente no contexto da solicitação, você deverá processar a chave conforme especificado na política. Se a chave não estiver presente, avalie o elemento da condição como verdadeiro.

Quando você usa `...IfExists` com uma chave de condição vazia, a condição corresponde somente se não houver chaves na solicitação. A AWS recomenda que, se você quiser testar se um contexto de solicitação está vazio, use o operador de condição `Null`.

- ...Operadores de condição ...IfExists
- Elementos de política JSON do IAM: Condition

Sugestão: Empty array principal (Entidade de matriz vazia)

Código do problema: EMPTY_ARRAY_PRINCIPAL

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Resolução da sugestão

Você deve usar o elemento `Principal` ou `NotPrincipal` nas políticas de confiança para funções do IAM e em políticas baseadas em recurso. As políticas baseadas em recursos são políticas que você incorpora diretamente em um recurso.

Quando você fornece uma matriz vazia em um elemento `Principal` de uma instrução, a instrução não tem efeito sobre a política. A AWS recomenda que você especifique as entidades principais que devem ter acesso ao recurso.

- Elementos de política JSON do IAM: entidade
- Elementos de política JSON do IAM: NotPrincipal

Sugestão: Empty array resource (Recurso de matriz vazio)

Código do problema: EMPTY_ARRAY_RESOURCE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Resolução da sugestão

As instruções devem incluir um elemento `Resource` ou `NotResource`.

Quando você fornece uma matriz vazia no elemento de recurso de uma instrução, a instrução não tem efeito sobre a política. A AWS recomenda que você especifique nomes de recursos da Amazon (ARNs) para os recursos.

- Elementos de política JSON do IAM: Resource
- Elementos de política JSON do IAM: NotResource

Sugestão: Empty object condition (Condição de objeto vazio)

Código do problema: EMPTY_OBJECT_CONDITION

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Resolução da sugestão

A estrutura do elemento `Condition` requer que você use um operador de condição e um par de chave-valor.

Quando você fornece um objeto vazio no elemento de condição de uma instrução, a instrução não tem efeito sobre a política. Remova o elemento opcional ou especifique condições.

- Elementos de política JSON do IAM: Condition

Sugestão: Empty object principal (Entidade de objeto vazia)

Código do problema: EMPTY_OBJECT_PRINCIPAL

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Resolução da sugestão

Você deve usar o elemento `Principal` ou `NotPrincipal` nas políticas de confiança para funções do IAM e em políticas baseadas em recurso. As políticas baseadas em recursos são políticas que você incorpora diretamente em um recurso.

Quando você fornece um objeto vazio no elemento `Principal` de uma instrução, a instrução não tem efeito sobre a política. A AWS recomenda que você especifique as entidades principais que devem ter acesso ao recurso.

- Elementos de política JSON do IAM: entidade principal
- Elementos de política JSON do IAM: NotPrincipal

Sugestão: Empty Sid value (Valor Sid vazio)

Código do problema: EMPTY_SID_VALUE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Resolução da sugestão

O elemento `Sid` (ID de instrução) opcional permite que você insira um identificador fornecido para a instrução de política. É possível atribuir um valor `Sid` a cada instrução em uma matriz de instruções. Se você optar por usar o elemento `Sid`, você deve fornecer um valor de string.

Termos relacionados

- Elementos de política JSON do IAM: Sid

Sugestão: equivalente a null false

Código de problema: EQUIVALENT_TO_NULL_FALSE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Equivalent to null false: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: false} to ensure better enforcement of the
 condition."
```

Resolução da sugestão

Substitua a chave de condição atual pela chave recomendada definida como `false`. Essa alteração torna a política mais clara e garante uma avaliação mais confiável das condições. Atualize seu bloco de condições para usar `{recommendedKey}: false` em vez da combinação de operador-chave atual.

Termos relacionados

- Elementos da política do IAM: condição
- Condições com várias chaves ou valores de contexto
- AWS Chaves de contexto de condições globais da

Sugestão: equivalente a null true

Código de problema: EQUIVALENT_TO_NULL_TRUE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Equivalent to null true: We recommend replacing the key {{key}} in the condition block
 of {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "We recommend replacing the key {{key}} in the condition block of
 {{operator}} with {{{recommendedKey}}: true} to ensure better enforcement of the
 condition."
```

Resolução da sugestão

Substitua a chave de condição atual pela chave recomendada definida como `true`. Essa alteração torna a política mais clara e garante uma avaliação mais confiável das condições. Atualize seu bloco de condições para usar `{recommendedKey}: true` em vez da combinação de operador-chave atual.

Termos relacionados

- Elementos da política do IAM: condição
- Condições com várias chaves ou valores de contexto
- AWS Chaves de contexto de condições globais da

Sugestão: Improve IP range (Melhorar o intervalo IP)

Código do problema: IMPROVE_IP_RANGE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Resolução da sugestão

As condições de endereço IP devem estar no formato CIDR padrão, como 203.0.113.0/24 ou 2001:DB8:1234:5678::/64. Quando você inclui bits diferentes de zero após os bits mascarados, eles não são considerados para a condição. A AWS recomenda que você use o novo endereço incluído na mensagem.

- Operadores de condição de endereço IP
- Elementos de política JSON do IAM: Condition

Sugestão: Null with qualifier (Nulo com qualificador)

Código do problema: NULL_WITH_QUALIFIER

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Resolução da sugestão

No elemento `Condition`, crie expressões em que você use operadores de condição, como igual ou menor que para comparar uma condição na política com chaves e valores no contexto da solicitação. Para solicitações que incluem vários valores para uma única chave de condição, você deve usar os operadores de conjunto `ForAllValues` ou `ForAnyValue`.

Quando você usa o operador de condição `Null` com`ForAllValues`, a instrução sempre retorna `true`. Quando você usa o operador de condição `Null` com`ForAnyValue`, a instrução sempre retorna `false`.A AWS recomenda o uso do operador de condição `StringLike` com esses operadores de conjunto.

Termos relacionados

- Chaves de contexto de múltiplos valores
- Operador de condição Null
- Elemento de condição

Sugestão: Private IP address subset (Subconjunto de endereços IP privados)

Código do problema: PRIVATE_IP_ADDRESS_SUBSET

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Resolução da sugestão

A chave de condição global `aws:SourceIp` funciona apenas para intervalos de endereços IP públicos.

Quando o elemento `Condition` incluir uma combinação de endereços IP privados e públicos, a instrução poderá não ter o efeito desejado. Você não pode especificar endereços IP privados usando `aws:VpcSourceIP`.

> **Note**
>
> A chave de condição global `aws:VpcSourceIP` será correspondente apenas se a solicitação se originar do endereço IP especificado e passar por um endpoint da VPC.

- Chave de condição global aws:SourceIp
- Chave de condição global aws:VpcSourceIp
- Operadores de condição de endereço IP
- Elementos de política JSON do IAM: Condition

Sugestão: Private NotIpAddress subset (Subconjunto NotipAddress privado)

Código do problema: PRIVATE_NOT_IP_ADDRESS_SUBSET

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Resolução da sugestão

A chave de condição global `aws:SourceIp` funciona apenas para intervalos de endereços IP públicos.

Quando o elemento `Condition` incluir o operador de condição `NotIpAddress` e uma combinação de endereços IP privados e públicos, a instrução poderá não ter o efeito desejado. Todos os endereços IP públicos não especificados na política serão correspondentes. Nenhum endereço IP privado será correspondente. Para obter esse efeito, você pode usar `NotIpAddress` com `aws:VpcSourceIP` e especificar os endereços IP privados que não devem ser correspondentes.

- Chave de condição global aws:SourceIp
- Chave de condição global aws:VpcSourceIp
- Operadores de condição de endereço IP
- Elementos de política JSON do IAM: Condition

Sugestão: Redundant action (Ação redundante)

Código do problema: REDUNDANT_ACTION

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Resolução da sugestão

Quando você usa curingas (*) no elemento `Action`, você pode incluir permissões redundantes. A AWS recomenda que você revise sua política e inclua somente as permissões necessárias. Isso pode ajudar você a remover ações redundantes.

Por exemplo, as ações a seguir incluem a ação `iam:GetCredentialReport` duas vezes.

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

Neste exemplo, as permissões são definidas para cada ação do IAM que começa com `Get` ou `List`. Quando o IAM adicionar operações adicionais de obtenção ou listagem, essa política as permitirá. É possível permitir todas essas ações somente leitura. A ação `iam:GetCredentialReport` já está incluída como parte de `iam:Get*`. Para remover as permissões duplicadas, você pode remover `iam:GetCredentialReport`.

Você recebe uma descoberta para esta verificação de política quando todo o conteúdo de uma ação é redundante. Neste exemplo, se o elemento incluir `iam:*CredentialReport`, ele não será considerado redundante. Isso inclui `iam:GetCredentialReport`, que é redundante, e `iam:GenerateCredentialReport`, que não é. A remoção de `iam:Get*` ou `iam:*CredentialReport` altera as permissões da política.

- Elementos de política JSON do IAM: Action

AWSPolíticas gerenciadas pela com esta sugestão

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

As ações redundantes não afetam as permissões concedidas pela política. Ao usar uma política gerenciada pela AWS como referência para criar sua política gerenciada pelo cliente, a AWS recomenda que você remova ações redundantes de sua política.

Sugestão: Redundant condition value num (Valor de condição redundante num)

Código do problema: REDUNDANT_CONDITION_VALUE_NUM

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Resolução da sugestão

Quando você usa operadores de condição numéricos para valores semelhantes em uma chave de condição, você pode criar uma sobreposição que resulte em permissões redundantes.

Por exemplo, o elemento `Condition` a seguir inclui várias condições `aws:MultiFactorAuthAge`que têm uma sobreposição de tempo de 1200 segundos.

```
"Condition": {
        "NumericLessThan": {
```

```
          "aws:MultiFactorAuthAge": [
            "2700",
            "3600"
          ]
        }
      }
```

Neste exemplo, as permissões são definidas se a autenticação multifator (MFA) tiver sido concluída há menos de 3600 segundos (1 hora). É possível remover o valor `2700` redundante.

- Operadores de condição numéricos
- Elementos de política JSON do IAM: Condition

Sugestão: Redundant resource (Recurso redundante)

Código do problema: REDUNDANT_RESOURCE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Resolução da sugestão

Ao usar curingas (*) em nomes de recursos da Amazon (ARN), você pode criar permissões de recurso redundantes.

Por exemplo, o elemento `Resource` a seguir inclui vários ARNs com permissões redundantes.

```
"Resource": [
```

```
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

Neste exemplo, as permissões são definidas para qualquer função com um nome que comece com `jane`. É possível remover os ARNs redundantes `jane-admin` e `jane-s3only` sem alterar as permissões resultantes. Isso torna a política dinâmica. Ele definirá permissões para quaisquer funções futuras que comecem com `jane`. Se a intenção da política for permitir o acesso a um número estático de funções, remova o último ARN e liste apenas os ARNs que devem ser definidos.

- Elementos de política JSON do IAM: Resource

AWSPolíticas gerenciadas pela com esta sugestão

As políticas gerenciadas pela AWS permitem que você comece a usar a AWS atribuindo permissões com base em casos de uso gerais da AWS.

Os recursos redundantes não afetam as permissões concedidas pela política. Ao usar uma política gerenciada pela AWS como referência para criar sua política gerenciada pelo cliente, a AWS recomenda que você remova recursos redundantes de sua política.

Sugestão: Redundant statement (Instrução redundante)

Código do problema: REDUNDANT_STATEMENT

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Resolução da sugestão

O elemento `Statement` é o principal elemento de uma política. Este elemento é obrigatório. O elemento `Statement` pode conter uma única instrução ou uma matriz de instruções individuais.

Quando você incluir a mesma instrução mais de uma vez em uma política longa, as instruções são redundantes. É possível remover uma das instruções sem afetar as permissões concedidas pela política. Quando alguém edita uma política, essa pessoa pode alterar uma das instruções sem atualizar a duplicação. Isso pode resultar em mais permissões do que o pretendido.

- Elementos de política JSON do IAM: Statement

Sugestão: Wildcard in service name (Curinga no nome do serviço)

Código do problema: WILDCARD_IN_SERVICE_NAME

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Resolução da sugestão

Ao incluir o nome de um produto da AWS em uma política, a AWS recomenda não incluir caracteres curinga (*,?). Isso pode adicionar permissões não pretendidas para serviços futuros. Por exemplo, há mais de dez produtos da AWS com a palavra `*code*` no nome.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- Elementos de política JSON do IAM: Resource

Sugestão: Allow with unsupported tag condition key for service (Permitir com chave de condição de etiqueta não suportada para o serviço)

Código do problema: ALLOW_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
 affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Resolução da sugestão

O uso de chaves de condição de etiqueta não suportadas no elemento `Condition` de uma política com `"Effect": "Allow"` não afeta as permissões concedidas pela política, porque a condição é ignorada para essa ação de serviço. A AWS recomenda que você remova as ações para serviços que não oferecem suporte à chave de condição e crie outra instrução para permitir o acesso a recursos específicos nesse serviço.

Se você usar a chave de condição `aws:ResourceTag` e uma ação de serviço não oferecer suporte a ela, a chave não será incluída no contexto da solicitação. Neste caso, a condição na instrução `Allow` sempre retorna `false` e a ação nunca é permitida. Isso acontece mesmo se o recurso estiver etiquetado corretamente.

Quando um serviço oferece suporte à chave de condição `aws:ResourceTag`, você pode usar etiquetas para controlar o acesso aos recursos desse serviço. Isso é conhecido como controle de

acesso baseado em atributo (ABAC). Os serviços que não oferecem suporte a essas chaves exigem que você controle o acesso a recursos usando o controle de acesso baseado em recursos (RBAC).

> **Note**
>
> Alguns serviços permitem oferecer suporte à chave de condição `aws:ResourceTag` para um subconjunto de seus recursos e ações. O IAM Access Analyzer retorna descobertas para as ações de serviço incompatíveis. Por exemplo, o Amazon S3 é compatível com `aws:ResourceTag` para um subconjunto de seus recursos. Para visualizar todos os tipos de recursos disponíveis no Amazon S3 que oferecem suporte à chave de condição `aws:ResourceTag`, consulte Tipos de recursos definidos pelo Amazon S3 na Referência de autorização do serviço.

Por exemplo, suponha que você deseja permitir que os membros da equipe exibam detalhes de recursos específicos etiquetados com o par de chave-valor `team=BumbleBee`. Suponha também que o AWS Lambda permita etiquetar recursos, mas não ofereça suporte à chave de condição `aws:ResourceTag`. Para permitir as ações de visualização para AWS App Mesh e AWS Backup se essa etiqueta estiver presente, use a chave de condição `aws:ResourceTag`. Para o Lambda, use uma convenção de nomenclatura de recursos que inclua o nome da equipe como o prefixo. Em seguida, inclua uma instrução separada que permita a visualização de recursos com essa convenção de nomenclatura.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
```

```
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

Warning

Não use a versão do operador de condição `Not` com `"Effect": "Allow"` como uma solução alternativa para essa descoberta. Esses operadores de condição fornecem correspondência negada. Isso significa que após a avaliação da condição, o resultado é negado. No exemplo anterior, a inclusão da ação `lambda:GetFunction` na instrução `AllowViewSupported` com o operador `StringNotEquals` sempre permite a ação, independentemente de o recurso estar ou não etiquetado.

Não use a versão ...`IfExists` do operador de condição como uma solução alternativa para essa descoberta. Isso significa “Permita a ação se a chave estiver presente no contexto da solicitação e os valores forem corresponderem. Caso contrário, permita a ação.” No exemplo anterior, a inclusão da ação `lambda:GetFunction` na instrução `AllowViewSupported` com o operador `StringEqualsIfExists` sempre permite a ação. Para essa ação, a chave não está presente no contexto e toda tentativa de visualizar esse tipo de recurso é permitida, independentemente de o recurso estar etiquetado.

Termos relacionados

- Chaves de condições globais
- Elementos de política JSON do IAM: operadores de condição
- Elemento de condição
- Visão geral das políticas de JSON

Sugestão: Allow NotAction with unsupported tag condition key for service (Permitir NotAction com chave de condição de etiqueta não suportada para o serviço)

Código do problema:
ALLOW_NOTACTION_WITH_UNSUPPORTED_TAG_CONDITION_KEY_FOR_SERVICE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Resolução da sugestão

O uso de chaves de condição de etiqueta não suportadas no elemento `Condition` de uma política com o elemento `NotAction` e `"Effect": "Allow"` não afeta as permissões concedidas pela política. A condição é ignorada para ações de serviço que não suportam a chave de condição. A AWS recomenda que você reescreva a lógica para permitir uma lista de ações.

Se você usar a chave de condição `aws:ResourceTag` com `NotAction`, quaisquer ações de serviço novas ou existentes que não ofereçam suporte à chave não serão permitidas. A AWS recomenda que você liste explicitamente as ações que você deseja permitir. O IAM Access Analyzer retorna uma descoberta separada para ações listadas que não oferecem suporte para a chave de condição `aws:ResourceTag`. Para obter mais informações, consulte Sugestão: Allow with unsupported tag condition key for service (Permitir com chave de condição de etiqueta não suportada para o serviço).

Quando um serviço oferece suporte à chave de condição `aws:ResourceTag`, você pode usar etiquetas para controlar o acesso aos recursos desse serviço. Isso é conhecido como controle de acesso baseado em atributo (ABAC). Os serviços que não oferecem suporte a essas chaves exigem que você controle o acesso a recursos usando o controle de acesso baseado em recursos (RBAC).

Termos relacionados

- Chaves de condições globais
- Comparação de ABAC com RBAC
- Elementos de política JSON do IAM: operadores de condição
- Elemento de condição
- Visão geral das políticas de JSON

Sugestão: chave de condição recomendada para a entidade principal de serviço

Código do problema: RECOMMENDED_CONDITION_KEY_FOR_SERVICE_PRINCIPAL

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Resolução da sugestão

Você pode especificar o Serviços da AWS no elemento `Principal` de uma política baseada em recursos usando uma entidade principal de serviço, que é um identificador para o serviço. Você deve usar as chaves de condições `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou

`aws:SourceOrgPaths` ao conceder acesso às entidades principais do serviço em vez de outras chaves de condições, como `aws:Referer`. Isso ajuda a evitar um problema de segurança chamado problema do substituto confuso.

Termos relacionados

- AWS service (Serviço da AWS) Entidades principais do
- AWS Chaves globais de condição da : aws:SourceAccount
- AWS Chaves globais de condição da : aws:SourceArn
- AWS Chaves de condições globais da : aws:SourceOrgId
- AWS Chaves de condições globais da : aws:SourceOrgPaths
- O problema "confused deputy

Sugestão: chave de condição irrelevante na política

Código do problema: IRRELEVANT_CONDITION_KEY_IN_POLICY

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Resolução da sugestão

Algumas chaves de condição não são relevantes para políticas baseadas em recursos. Por exemplo, a chave de condição `s3:ResourceAccount` não é relevante para a política baseada em recursos anexada a um bucket do Amazon S3 ou para o tipo de recurso de ponto de acesso do Amazon S3.

É possível usar a chave de condição em uma política baseada em identidade para controlar o acesso aos recurso.

Termos relacionados

- Identity-based policies and resource-based policies

Sugestão: chave redundante devido ao curinga na condição

Código de problema: REDUNDANT_KEY_DUE_TO_WILDCARD_IN_CONDITION

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant key due to wildcard in condition: The key {{key}} in the condition block of
 {{operator}} is redundant because it is always matched. Remove this key to simplify
 the condition.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} is redundant
 because it is always matched. Remove this key to simplify the condition."
```

Resolução da sugestão

Remova a chave de condição redundante da política. Sempre existe uma correspondência para a chave por causa do padrão curinga, o que a torna desnecessária. Simplifique seu bloco de condições removendo essa chave enquanto mantém as mesmas permissões em vigor.

Termos relacionados

- Elementos da política do IAM: condição
- Condições com várias chaves ou valores de contexto
- AWS Chaves de contexto de condições globais da

Sugestão: entidade principal redundante na política de confiança da função

Código do problema: REDUNDANT_PRINCIPAL_IN_ROLE_TRUST_POLICY

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Resolução da sugestão

Se você especificar tanto uma entidade principal da função assumida quanto a função superior a ela no elemento `Principal` de uma política, ela não permite nem nega nenhuma permissão diferente. Por exemplo, haverá redundância se você especificar o elemento `Principal` usando o seguinte formato:

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Recomendamos remover a entidade principal da função assumida.

Termos relacionados

- Entidades principais da sessão de função

Sugestão: instrução redundante devido ao curinga na condição

Código de problema: REDUNDANT_STATEMENT_DUE_TO_WILDCARD_IN_CONDITION

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Redundant statement due to wildcard in condition: The key {{key}} in the condition
 block of {{operator}} does not match any values. Remove this key to simplify the
 condition.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The key {{key}} in the condition block of {{operator}} does not
 match any values. Remove this key to simplify the condition."
```

Resolução da sugestão

Remova a chave de condição que não corresponde a nenhum valor. Essa chave cria uma condição inalcançável que nunca será satisfeita, tornando-a redundante. Limpe a política removendo essa chave para melhorar a legibilidade e a performance.

Termos relacionados

- Elementos da política do IAM: condição
- Condições com várias chaves ou valores de contexto
- AWS Chaves de contexto de condições globais da

Sugestão: confirme o tipo de reivindicação do público

Código do problema: CONFIRM_AUDIENCE_CLAIM_TYPE

Tipo de descoberta: SUGGESTION

Detalhes da descoberta

No Console de gerenciamento da AWS, a descoberta desta verificação inclui a seguinte mensagem:

```
Confirm audience claim type: The "{{key}}" ({{audienceType}}) claim key identifies
 the recipients that the JSON web token is intended for. Because this claim is single-
valued, do not use a qualifier.
```

Em chamadas programáticas para a AWS CLI ou a API da AWS, a descoberta para esta verificação inclui a seguinte mensagem:

```
"findingDetails": "The "{{key}}" ({{audienceType}}) claim key identifies the recipients
 that the JSON web token is intended for. Because this claim is single-valued, do not
 use a qualifier."
```

Resolução da sugestão

A chave de solicitação (público) `aud` é um identificador exclusivo da aplicação, que é emitida quando você registra a aplicação no IdP e identifica os destinatários aos quais o token da Web JSON s destina. As reivindicações do público podem ser de valor único ou de vários valores. Se a reivindicação for de vários valores, use um operador de conjunto de condições `ForAllValues` ou `ForAnyValue`. Se a reivindicação for de valor único, não use um operador de conjunto de condições.

Termos relacionados

- Criar uma função para identidade da Web ou federação do OpenID Connect (console)
- Chaves de contexto de múltiplos valores
- Chaves de condição de valor único comparadas com chaves de condição de vários valores

Validar políticas com verificações de políticas do IAM Access Analyzer

Você pode usar verificações de políticas personalizadas para verificar novos acessos com base em seus padrões de segurança. Uma cobrança é associada a cada verificação de novo acesso. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Validar políticas com verificações personalizadas de políticas (console)

Como uma etapa opcional, você pode executar uma verificação personalizada de política ao editar uma política no editor de políticas de JSON no console do IAM. É possível verificar se a política atualizada concede novo acesso em comparação com a versão existente.

Para verificar se há novos acessos ao editar políticas JSON do IAM

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Policies (Políticas).
3. Na lista de políticas, escolha o nome da política que deseja visualizar. É possível usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha a guia Permissões e depois escolha Editar.
5. Escolha a opção JSON e atualize sua política.
6. No painel de validação de política abaixo da política, escolha a guia Verificar novos acessos e, em seguida, escolha Verificar política. Se as permissões modificadas concederem novo acesso, a declaração será destacada no painel de validação da política.
7. Se você não pretende conceder um novo acesso, atualize a declaração de política e escolha Verificar política até que nenhum novo acesso seja detectado.

   > **Note**
   >
   > Uma cobrança é associada a cada verificação de novo acesso. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

8. Escolha Próximo.
9. Na página Revisar e salvar, revise Permissões definidas nessa política e escolha Salvar alterações.

Validar políticas com verificações personalizadas de políticas (AWS CLI ou API)

Você pode executar verificações de políticas personalizadas do IAM Access Analyzer a partir da AWS CLI ou da API do IAM Access Analyzer.

Para executar verificações personalizadas de política (AWS CLI) do IAM Access Analyzer

- Para verificar se um novo acesso é permitido para uma política atualizada em comparação com a política existente, execute o seguinte comando: check-no-new-access
- Para verificar se o acesso especificado não é permitido por uma política, execute o seguinte comando: check-access-not-granted

- Para verificar se uma política de recursos pode conceder acesso público a um tipo de recurso especificado, execute o seguinte comando: check-no-public-access

Verificações personalizadas de política do IAM Access Analyzer (API)

- Para verificar se um novo acesso é permitido para uma política atualizada em comparação com a política existente, use a operação da API CheckNoNewAccess.
- Para verificar se o acesso especificado não é permitido por uma política, use a operação da API CheckAccessNotGranted.
- Para verificar se uma política de recursos pode conceder acesso público a um tipo de recurso especificado, execute a operação CheckNoPublicAccess de API.

Geração de política do IAM Access Analyzer

Como administrador ou desenvolvedor, você pode conceder permissões a entidades do IAM (usuários ou funções) além do que elas exigem. O IAM fornece várias opções para ajudar você a refinar as permissões concedidas.

Uma opção é gerar uma política do IAM baseada na atividade de acesso para uma entidade. O IAM Access Analyzer revisa seus logs do AWS CloudTrail e gera um modelo de política que contém as permissões que a entidade usou no intervalo de datas especificado. Você pode usar o modelo para criar uma política com permissões refinadas que concedem apenas as permissões detalhadas para dar suporte ao seu caso de uso específico.

Tópicos

- Como funciona a geração de políticas
- Informações em nível de serviço e ação
- O que é necessário saber para gerar políticas
- Permissões necessárias para gerar uma política
- Gerar uma política com base na atividade do CloudTrail (console)
- Gerar uma política usando dados do AWS CloudTrail em outra conta
- Gerar uma política com base na atividade do CloudTrail (AWS CLI)
- Gerar uma política com base na atividade do CloudTrail (API da AWS)
- Serviços de geração de política do IAM Access Analyzer

Como funciona a geração de políticas

IAM Access Analyzer analisa seus eventos do CloudTrail para identificar ações e serviços usados por uma entidade (usuário ou função) do IAM. Em seguida, gera uma política do IAM com base nessa atividade. Você pode refinar as permissões de uma entidade ao substituir uma política de permissões ampla anexada à entidade pela política gerada. Veja a seguir uma visão geral de alto nível do processo de geração de políticas.

- Set up for policy template generation (Configurar a geração de modelo de política): você especifica um período de tempo de até 90 dias para o IAM Access Analyzer analisar seus histórico de eventos do AWS CloudTrail. Você deve especificar um perfil de serviço existente ou criar um novo. O perfil de serviço dá ao IAM Access Analyzer acesso a sua trilha do CloudTrail e informações do último serviço acessado para identificar os serviços e ações que foram usados. Você deve especificar a trilha do CloudTrail que está registrando os evento de logs na conta antes de gerar uma política. Para obter mais informações sobre cotas do IAM Access Analyzer de dados do CloudTrail, consulte Cotas do IAM Access Analyzer.
- Gerar política – IAM Access Analyzer gera uma política com base na atividade de acesso em seus eventos do CloudTrail.
- Revisar e personalizar a política – Depois que a política for gerada, você poderá revisar os serviços e ações usados pela entidade durante o intervalo de datas especificado. Você pode personalizar ainda mais a política adicionando ou removendo permissões, especificando recursos e adicionando condições ao modelo de política.
- Criar e anexar uma política – Você tem a opção de salvar a política gerada com a criação de uma política gerenciada. Você pode anexar a política criada ao usuário ou função cuja atividade foi usada para gerar a política.

Informações em nível de serviço e ação

Quando IAM Access Analyzer gera uma política do IAM, as informações são retornadas para ajudar você a personalizar ainda mais a política. Duas categorias de informações podem ser retornadas quando uma política é gerada:

- Policy with action-level information (Política com informações no nível de ação): para alguns produtos da AWS, como o Amazon EC2, o IAM o Access Analyzer podem identificar as ações encontradas nos eventos do CloudTrail e listar as ações usadas na política gerada. Para obter uma lista dos serviços compatíveis, consulte Serviços de geração de política do IAM Access

Analyzer. Para alguns serviços, o IAM Access Analyzer solicita que você adicione ações para os serviços à política gerada.

- A política com informações em nível de serviço – IAM Access Analyzer usa as últimas informações acessadas para criar um modelo de política com todos os serviços usados recentemente. Ao usar o Console de gerenciamento da AWS, solicitamos que você revise os serviços e adicione ações para concluir a política.

Para obter uma lista de ações em cada serviço, consulte Ações, recursos e chaves de condição para produtos da AWS na Referência de autorização do serviço.

O que é necessário saber para gerar políticas

Antes de gerar uma política, analise os seguintes detalhes importantes.

- Habilitar uma trilha do CloudTrail – Você deve ter uma trilha do CloudTrail habilitada na sua conta para gerar uma política com base na atividade de acesso. Ao criar uma trilha do CloudTrail, o CloudTrail envia eventos relacionados à sua trilha para um bucket do Amazon S3 especificado por você. Para saber como criar uma trilha do CloudTrail, consulte Criar uma trilha para a conta da AWS no Guia do usuário do AWS CloudTrail.
- Data events not available (Eventos de dados não disponíveis): o IAM Access Analyzer não identifica a atividade no nível de ação para eventos de dados, como eventos de dados do Amazon S3, em políticas geradas.
- PassRole – A ação `iam:PassRole` não é rastreada pelo CloudTrail e não é incluída nas políticas geradas.
- Reduzir o tempo de geração de políticas – Para gerar uma política mais rapidamente, reduza o intervalo de datas especificado durante a configuração da geração de políticas.
- Usar CloudTrail para auditoria – Não use a geração de políticas para fins de auditoria; em vez disso, use CloudTrail. Para obter mais informações sobre o uso do CloudTrail, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.
- Ações negadas: a geração de políticas analisa todos os eventos do CloudTrail, inclusive as ações negadas.
- Uma política por console de IAM – Você pode gerar uma política por vez no console de IAM.
- Console de IAM de disponibilidade de políticas geradas – Você pode analisar uma política gerada no console de IAM até sete dias após ela ter sido gerada. Após sete dias, você deverá gerar uma nova política.

- Policy generation quotas (Cotas de geração de políticas): para obter mais informações sobre cotas de geração de políticas do IAM Access Analyzer, consulte Cotas do IAM Access Analyzer.
- As tarifas do Amazon S3 Standard se aplicam: quando você usa o atributo de geração de políticas, o IAM Access Analyzer analisa os logs do CloudTrail no bucket do S3. Não há cobranças adicionais de armazenamento para acessar os logs do CloudTrail para geração de políticas. A AWS cobra as tarifas padrão do Amazon S3 para solicitações e transferência de dados de logs do CloudTrail armazenados no bucket do S3.
- Suporte do AWS Control Tower: a geração de políticas não oferece suporte ao uso de trilhas do AWS CloudTrail criadas pelo AWS Control Tower ao gerar políticas pelos seguintes motivos:
  - Os dados do CloudTrail da organização são registrados em log em outra conta, a conta de arquivamento de log do AWS Control Tower.
  - As permissões para o bucket do S3 em que esses logs são armazenados não podem ser reconfiguradas devido às restrições no bucket de registro do S3 definidas pelas políticas de controle de serviço (SCPs) da AWS Control Tower.

Permissões necessárias para gerar uma política

As permissões que você precisa para gerar uma política pela primeira vez são diferentes daquelas necessárias para gerar uma política para usos subsequentes. Para obter mais informações, consulte Conceitos básicos do AWS Identity and Access Management Access Analyzer.

Configuração pela primeira vez

Ao gerar uma política pela primeira vez, você deve escolher um perfil de serviço existente adequado em sua conta ou criar um novo perfil de serviço. O perfil de serviço dá ao IAM Access Analyzer acesso ao CloudTrail e às informações do último serviço acessado em sua conta. Apenas os administradores devem ter as permissões necessárias para criar e configurar funções. Portanto, recomendamos que um administrador crie o perfil de serviço durante a primeira configuração. Para saber mais sobre as permissões necessárias para criar perfis de serviço, consulte Criar uma função para delegar permissões a um produto da AWS.

Permissões necessárias para o perfil de serviço

Ao criar um perfil de serviço, você configura duas políticas para o perfil. Você anexa uma política de permissões do IAM à função que especifica o que a função pode fazer. Você também anexa uma política de confiança da função que especifica a entidade principal que pode usar a função.

O primeiro exemplo de política mostra a política de permissões para o perfil de serviço necessário para gerar uma política. O segundo exemplo de política mostra a política de confiança do perfil que é necessário para o perfil de serviço. Você pode usar essas políticas para ajudar na criação de um perfil de serviço ao usar a API AWS ou AWS CLI para gerar uma política. Ao usar o console do IAM para criar um perfil de serviço como parte do processo de geração de políticas, nós geramos essas políticas para você.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

O exemplo a seguir mostra a política de confiança da função com as permissões que permitem que o IAM Access Analyzer assuma a função.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Usos subsequentes

Para gerar políticas no Console de gerenciamento da AWS, um usuário do IAM deve ter uma política de permissões que permita transmitir o perfil de serviço usado para geração de políticas para o IAM Access Analyzer. A ação `iam:PassRole` geralmente é acompanhada por `iam:GetRole` para que o usuário possa obter os detalhes do perfil a ser transmitido. Neste exemplo, o usuário pode transmitir apenas funções que existam na conta especificada com nomes que comecem com : `AccessAnalyzerMonitorServiceRole*`. Para saber mais sobre como passar perfis do IAM para produtos da AWS, consulte Conceder permissões a um usuário para passar um perfil para um produto da AWS.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
```

```
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

Você deve ter as seguintes permissões do IAM Access Analyzer para gerenciar políticas no Console de gerenciamento da AWS, na API da AWS ou na AWS CLI, conforme mostrado na seguinte declaração de política.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

Para usos pela primeira vez e subsequentes

Quando você usa o Console de gerenciamento da AWS para gerar uma política, deve ter a permissão `cloudtrail:ListTrails` para listar as trilhas do CloudTrail em sua conta, conforme mostrado na instrução de política a seguir.

```
{
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "cloudtrail:ListTrails"
  ],
  "Resource": "*"
}
```

Gerar uma política com base na atividade do CloudTrail (console)

Você pode gerar uma política para um usuário ou perfil do IAM.

Etapa 1: gerar uma política com base na atividade do CloudTrail

O procedimento a seguir explica como gerar uma política para uma função usando o Console de gerenciamento da AWS.

Gerar uma política para um perfil do IAM

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Roles (Funções).

> **Note**
>
> As etapas para gerar uma política com base na atividade de um usuário do IAM são quase idênticas. Para fazer isso, escolha Users(Usuários) em vez de Roles (Funções).

3. Na lista de funções na sua conta, escolha o nome da função cuja atividade você deseja usar para gerar uma política.
4. Na guia Permissions (Permissões), na seção Gerar políticas com base em eventos do CloudTrail, escolha Generate policy (Gerar política).
5. Na página Generate policy (Gerar política), especifique o período em que IAM Access Analyzerdeseja analisar seus eventos do CloudTrail para ações realizadas com a função. Você pode escolher um intervalo de até 90 dias. Recomendamos que escolha o menor período possível para reduzir o tempo de geração de políticas.
6. Na seção CloudTrail access (Acesso ao CloudTrail), escolha uma função existente adequada ou crie uma nova função se não existir uma função adequada. A função concede ao IAM Access Analyzer permissões para acessar seus dados do CloudTrail em seu nome para revisar a atividade de acesso e identificar os serviços e ações que foram usados. Para saber mais sobre as permissões necessárias para essa função, consulte Permissões necessárias para gerar uma política.
7. Na seção CloudTrail trilha a ser analisada, especifique a trilha do CloudTrail que registra eventos de logs para a conta.

Se você escolher uma trilha do CloudTrail que armazene logs em outra conta, uma caixa de informações sobre o acesso entre contas será exibida. O acesso entre contas requer configuração adicional. Para saber mais, consulte Choose a role for cross-account access mais adiante neste tópico.

8. Escolha Generate policy (Gerar política).
9. Enquanto a geração de políticas está em andamento, você é levado de volta à página Roles Summary (Resumo das funções) na guia Permissions (Permissões). Aguarde até que o status na seção Detalhes da solicitação de política exiba Success(Sucesso) e escolha View generated policy (Exibir política gerada). Você pode visualizar a política gerada por até sete dias. Se você gerar outra política, ela substituirá a política existente.

Etapa 2: revisar permissões e adicionar ações para serviços usados

Revise os serviços e ações que o IAM Access Analyzer identificou que foram usados pela função. Você pode adicionar ações para todos os serviços usados ao modelo de política gerado.

1. Analise as seguintes seções:
   - Na página Review permissions (Revisar permissões), analise a lista de Actions included in the generated policy (Ações incluídas na política gerada). A lista exibe os serviços e as ações que o IAM Access Analyzer identificou que foram usadas pela função no intervalo de datas especificado.
   - A seção Services used (Serviços usados) exibe outros serviços que o IAM Access Analyzer identificou que foram usados pela função no intervalo de datas especificado. As informações sobre quais ações foram usadas podem não estar disponíveis para os serviços listados nesta seção. Use os menus de cada serviço listado para escolher manualmente as ações que deseja incluir na política.
2. Quando terminar de adicionar ações, escolha Next (Avançar).

Etapa 3: personalizar ainda mais a política gerada

Você pode personalizar ainda mais a política adicionando ou removendo permissões ou especificando recursos.

Para personalizar a política gerada

1. Atualize o modelo de política. O modelo da política contém espaços reservados do ARN do recurso para ações que oferecem suporte a permissões em nível de recurso, segundo as indicações da imagem a seguir. Permissões no nível do recurso se referem à capacidade de especificar em quais recursos os usuários têm permissão para realizar ações. Recomendamos que você use ARNs para especificar os recursos individuais na política para ações que oferecem suporte a permissões em nível do recurso. Você pode substituir os espaços reservados de ARNs do recurso por ARNs de recurso válidos para o seu caso de uso.

   Se uma ação não for compatível com permissões em nível do recurso, você deverá usar um curinga * para especificar que todos os recursos podem ser afetados pela ação. Para saber quais produtos da AWS oferecem suporte a permissões no nível de recurso, consulte Produtos da AWS que funcionam com o IAM. Para obter uma lista de ações em cada serviço e saber quais ações são compatíveis com permissões em nível do recurso, consulte Ações, recursos e chaves de condição para serviços da AWS.

2. (Opcional) Adicione, modifique ou remova declarações de política JSON no modelo. Para saber mais sobre como escrever políticas de JSON, consulte Criar políticas do IAM (console).
3. Quando você terminar de personalizar o modelo de política, terá as seguintes opções:

   - (Opcional) Você pode copiar o JSON no modelo para usar separadamente fora da página Generated policy (Política gerada). Por exemplo, se você quiser usar o JSON para criar

uma política em uma conta diferente. Se a política em seu modelo exceder o limite de 6.144 caracteres das políticas JSON, ela será dividida em várias políticas.

- Escolha Next (Próximo) para analisar e criar uma política gerenciada na mesma conta.

Etapa 4: revisar e criar uma política gerenciada

Se você tiver permissões para criar e anexar políticas do IAM, poderá criar uma política gerenciada a partir da política gerada. Em seguida, você poderá anexar a política a um usuário ou função na sua conta.

Para analisar e criar uma política

1. Na página Review and create managed policy (Revisar e criar uma política gerenciada), digite um Name (Nome) e uma Description (Descrição) (opcional) para a política que você está criando.
2. (Opcional) Na seção Summary (Resumo), você pode analisar as permissões que serão incluídas na política.
3. (Opcional) Adicione metadados à política associando tags como pares de chave-valor. Para obter mais informações sobre como usar etiquetas no IAM, consulte Recursos de etiquetas do IAM.
4. Quando terminar, execute uma as seguintes ações:

   - Você pode anexar a nova política diretamente à função que foi usada para gerar a política. Para fazer isso, perto da parte inferior da página, marque a caixa de seleção ao lado de Anexar política ao ***YourRoleName***. Em seguida, escolha Create and attach policy (Criar e anexar política).
   - Caso contrário, escolha Create policy (Criar política). Você pode encontrar a política criada na lista de políticas no painel de navegação Policies (Políticas) do console do IAM.
5. Você pode anexar a política criada a uma entidade em sua conta. Depois de anexar a política, você pode remover qualquer outra política excessivamente ampla que possa estar anexada à entidade. Para saber como anexar uma política gerenciada, consulte Adicionar permissões de identidade do IAM (console).

Gerar uma política usando dados do AWS CloudTrail em outra conta

Você pode criar trilhas do CloudTrail que armazenam dados em contas centrais para simplificar as atividades de governança. Por exemplo, você pode usar o AWS Organizations para criar uma trilha que registre todos os eventos para todas as Contas da AWS nessa organização. A trilha pertence a uma conta central.

Se você quiser gerar uma política para um usuário ou uma função em uma conta diferente da conta em que seus dados de log do CloudTrail estão armazenados, você deve conceder acesso entre contas. Para fazer isso, você precisa de uma função e uma política de bucket que concedam permissões do IAM Access Analyzer aos logs do CloudTrail. Para obter mais informações sobre como criar trilhas do Organizations, consulte Criar uma trilha para uma organização.

Neste exemplo, suponha que você deseje gerar uma política para um usuário ou uma função na conta A. A trilha do CloudTrail na conta A armazena logs do CloudTrail em um bucket na conta B. Antes de gerar uma política, você deve fazer as seguintes atualizações:

1. Escolha um perfil existente ou crie um novo perfil de serviço que conceda ao IAM Access Analyzer acesso ao bucket na conta B (onde seus logs do CloudTrail estão armazenados).
2. Atualize a política de propriedade de objetos de bucket e permissões de bucket do Amazon S3 na conta B para permitir que o IAM Access Analyzer acesse objetos no bucket.

Etapa 1: Escolher ou criar uma função para acesso entre contas

- Na tela Generate policy (Gerar política), a opção Use an existing role (Usar uma função existente) é pré-selecionada para você se existir uma função com as permissões necessárias em sua conta. Caso contrário, escolha Create and use a new service role (Criar e usar um novo perfil de serviço). A nova função é usada para conceder acesso do IAM Access Analyzer aos logs do CloudTrail na conta B.

Etapa 2: Verificar ou atualizar a configuração de bucket do Amazon S3 na conta B

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
2. Na lista Buckets, escolha o nome do bucket onde seus logs de trilha do CloudTrail estão armazenados.

3. Selecione a guia Permissions (Permissões) e localize a seção Object ownership (Propriedade de objeto).

   Use as configurações de bucket de Amazon S3 Object Ownership (Propriedade de objetos do Amazon S3) para controlar a propriedade de objetos que são carregados nos buckets. Por padrão, quando outras Contas da AWS carregam objetos no seu bucket, os objetos pertencem à conta que faz o upload. Para gerar uma política, todos os objetos do bucket devem pertencer ao proprietário do bucket. Dependendo do caso de uso de ACL, talvez seja necessário alterar a configuração Object Ownership (Propriedade do objeto) para o bucket. Defina Object Ownership (Propriedade do objeto) como uma das opções a seguir.

   - Bucket owner enforced (Proprietário do bucked obrigatório) (recomendado)
   - Bucket owner preferred (Proprierário do bucket preferencial

   > **Important**
   >
   > Para gerar uma política com sucesso, os objetos do bucket devem pertencer ao proprietário do bucket. Se escolher Bucket owner preferred (Proprierário do bucket preferencial), você só poderá gerar uma política para o período de tempo após a alteração da propriedade do objeto.

   Para saber mais sobre propriedade de objetos no Amazon S3, consulte Controlar a propriedade de objetos e desabilitar ACLs para seu bucket no Guia do Usuário do Amazon S3.

4. Adicione permissões à política de bucket do Amazon S3 na conta B para permitir acesso à função na conta A.

   A política de exemplo a seguir permite `ListBucket` e `GetObject` para o bucket chamado `amzn-s3-demo-bucket`. Ela permitirá o acesso se a função que acessa o bucket pertencer a uma conta em sua organização e tiver um nome que comece com `AccessAnalyzerMonitorServiceRole`. Usar `aws:PrincipalArn` como uma `Condition` no elemento `Resource` garante que o perfil só possa acessar a atividade da conta se ela pertencer à conta A. É possível substituir `amzn-s3-demo-bucket` pelo nome do bucket, `optional-prefix` por um prefixo opcional para o bucket e `organization-id` pelo ID da organização.

   ```
   {
   ```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. Se você criptografar logs usando o AWS KMS, atualize a política de chave do AWS KMS na conta em que você armazena os logs do CloudTrail para conceder ao IAM Access Analyzer acesso para usar a sua chave, conforme mostrado no exemplo de política a seguir. Substitua `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` pelo ARN da sua trilha e `organization-id` pelo ID da sua organização.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
```

```
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
            ]
        },
        "ArnLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
     }
    ]
}
```

Gerar uma política com base na atividade do CloudTrail (AWS CLI)

Você pode usar os seguintes comandos para gerar uma política usando o AWS CLI.

Para gerar uma política

- aws accessanalyzer start-policy-generation

Para exibir uma política gerada

- aws accessanalyzer get-generated-policy

Para cancelar uma solicitação de geração de política

- aws accessanalyzer cancel-policy-generation

Para exibir uma lista de solicitações de geração de políticas

- aws accessanalyzer list-policy-generations

Gerar uma política com base na atividade do CloudTrail (API da AWS)

Você pode usar as seguintes operações para gerar uma política usando a API AWS.

Para gerar uma política

- StartPolicyGeneration

Para exibir uma política gerada

- GetGeneratedPolicy

Para cancelar uma solicitação de geração de política

- CancelPolicyGeneration

Para exibir uma lista de solicitações de geração de políticas

- ListPolicyGenerations

Serviços de geração de política do IAM Access Analyzer

A tabela a seguir lista os serviços da AWS para os quais o IAM Access Analyzer gera políticas com informações de nível de ação. Para obter uma lista de ações em cada serviço, consulte Ações, recursos e chaves de condição dos serviços da AWS na Referência de autorização do serviço.

AWSA fornece informações de geração de política no formato JSON para agilizar a automação dos fluxos de trabalho de gerenciamento de políticas. Com as informações de referência do serviço, você pode acessar informações de geração de política de acesso em Serviços da AWS de arquivos legíveis por máquina. Para obter mais informações, consulte Simplified AWS service (Serviço da AWS) information for programmatic access na Service Authorization Reference.

Serviço	Prefixo do serviço
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Gerenciamento de contas	conta
Gerenciador de certificados da AWS	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Application Cost Profiler	application-cost-profiler
Amazon CloudWatch Application Insights	applicationinsights
AWS App Mesh	appmesh
Amazon WorkSpaces Applications	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	ajuste de escala automático
AWS Marketplace	aws-marketplace

Serviço	Prefixo do serviço
AWS Backup	backup
AWS Batch	lote
Amazon Braket	braket
AWS Budgets	orçamentos
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
Amazon CodeGuru Reviewer	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Notificações do	codestar-notifications
Identidade do Amazon Cognito	cognito-identity
Grupos de usuários do Amazon Cognito	cognito-idp

Serviço	Prefixo do serviço
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect Customer	conectar
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Amazon DocumentDB Elastic Clusters	docdb-elastic
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr

Serviço	Prefixo do serviço
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon EMR em EKS (EMR Containers)	emr-containers
Amazon EMR Sem Servidor	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	eventos
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift Servers	gamelift
Amazon Location Service	geo
Amazon Glacier	glacier

Serviço	Prefixo do serviço
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS Armazenamento de identidades do	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs

Serviço	Prefixo do serviço
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex V2	lex
AWS License Manager Linux Subscriptions Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	logs
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive

Serviço	Prefixo do serviço
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB	memorydb
AWS Transform MGN	mgn
AWS Migration Hub	mgh
AWS Migration Hub Strategy Recommendations	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organizações
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Perfis de clientes do Amazon Connect Customer	perfil
Amazon QLDB	qldb
AWS Resource Access Manager	ram

Serviço	Prefixo do serviço
AWS Lixeira da	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de dados do Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorador de recursos da AWS	resource-explorer-2
Grupos de recursos da AWS	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Roles Anywhere	rolesanywhere
Amazon Route 53	route53
Controles de recuperação do Amazon Route	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3
Amazon S3 on Outposts	s3-outposts

Serviço	Prefixo do serviço
Recursos geoespaciais do Amazon SageMaker AI	sagemaker-geospatial
Savings Plans	savingsplans
Esquemas do Amazon EventBridge	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub CSPM	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
SMS e serviço de voz do Amazon Pinpoint	sms-voice
AWS Snowball Edge	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm

Serviço	Prefixo do serviço
AWS Systems Manager Incident Manager	ssm-incidents
AWS Systems Manager para SAP	ssm-sap
AWS Step Functions	estados
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	tag
Amazon Textract	textract
Amazon Timestream	timestream
AWS Telco Network Builder	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transferência
Amazon Translate	translate
ID de voz do Connect Customer	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Customer Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaços de trabalho

Serviço	Prefixo do serviço
AWS X-Ray	xray

Ações para informações sobre geração de políticas

A tabela a seguir lista os as ações para as quais o IAM Access Analyzer gera políticas com informações de nível de ação.

> **Important**
>
> A ação `iam:UpdateAccountName` será descontinuada em 22 de abril de 2026. Depois de 22 de abril de 2026, somente a permissão `account:PutAccountName` controlará o acesso à atualização do nome da conta. É altamente recomendável que você atualize todas as políticas de controle de serviços (SCPs) que controlam as atualizações do nome da conta para usar a permissão `account:PutAccountName`.

Prefixo do serviço	Ações do
access-analyzer	access-analyzer:ApplyArchiveRule
	access-analyzer:CancelPolicyGeneration
	access-analyzer:CheckAccessNotGranted
	access-analyzer:CheckNoNewAccess
	access-analyzer:CheckNoPublicAccess
	access-analyzer:CreateAccessPreview
	access-analyzer:CreateAnalyzer
	access-analyzer:CreateArchiveRule
	access-analyzer:DeleteAnalyzer
	access-analyzer:DeleteArchiveRule

Prefixo do serviço	Ações do
	access-analyzer:GenerateFindingRecommendation
	access-analyzer:GetAccessPreview
	access-analyzer:GetAnalyzedResource
	access-analyzer:GetAnalyzer
	access-analyzer:GetArchiveRule
	access-analyzer:GetFinding
	access-analyzer:GetFindingRecommendation
	access-analyzer:GetFindingsStatistics
	access-analyzer:GetGeneratedPolicy
	access-analyzer:ListAccessPreviewFindings
	access-analyzer:ListAccessPreviews
	access-analyzer:ListAnalyzedResources
	access-analyzer:ListAnalyzers
	access-analyzer:ListArchiveRules
	access-analyzer:ListFindings
	access-analyzer:ListPolicyGenerations
	access-analyzer:StartPolicyGeneration
	access-analyzer:StartResourceScan
	access-analyzer:UpdateAnalyzer
	access-analyzer:UpdateArchiveRule
	access-analyzer:UpdateFindings

Prefixo do serviço	Ações do
	access-analyzer:ValidatePolicy
conta	account:AcceptPrimaryEmailUpdate account:DeleteAlternateContact account:DisableRegion account:EnableRegion account:GetAccountInformation account:GetAlternateContact account:GetContactInformation account:GetGovCloudAccountInformation account:GetPrimaryEmail account:GetRegionOptStatus account:ListRegions account:PutAccountName account:PutAlternateContact account:PutContactInformation account:StartPrimaryEmailUpdate

Prefixo do serviço	Ações do
acm	acm:CreateAcmeEndpoint
	acm:DeleteCertificate
	acm:DescribeCertificate
	acm:ExportCertificate
	acm:GetAccountConfiguration
	acm:GetCertificate
	acm:ImportCertificate
	acm:ListAcmeEndpoints
	acm:ListCertificates
	acm:PutAccountConfiguration
	acm:RenewCertificate
	acm:RequestCertificate
	acm:ResendValidationEmail
	acm:SearchCertificates
	acm:UpdateCertificateOptions

Prefixo do serviço	Ações do
airflow	airflow:CreateCliToken airflow:CreateEnvironment airflow:CreateWebLoginToken airflow:DeleteEnvironment airflow:GetEnvironment airflow:ListEnvironments airflow:PublishMetrics airflow:UpdateEnvironment

Prefixo do serviço	Ações do
amplify	amplify:CreateApp
	amplify:CreateBackendEnvironment
	amplify:CreateBranch
	amplify:CreateDeployment
	amplify:CreateDomainAssociation
	amplify:CreateWebHook
	amplify:DeleteApp
	amplify:DeleteBackendEnvironment
	amplify:DeleteBranch
	amplify:DeleteDomainAssociation
	amplify:DeleteJob
	amplify:DeleteWebHook
	amplify:GenerateAccessLogs
	amplify:GetApp
	amplify:GetArtifactUrl
	amplify:GetBackendEnvironment
	amplify:GetBranch
	amplify:GetDomainAssociation
	amplify:GetJob
	amplify:GetWebHook
	amplify:ListApps

Prefixo do serviço	Ações do
	amplify:ListArtifacts
	amplify:ListBackendEnvironments
	amplify:ListBranches
	amplify:ListDomainAssociations
	amplify:ListJobs
	amplify:ListWebHooks
	amplify:StartDeployment
	amplify:StartJob
	amplify:StopJob
	amplify:UpdateApp
	amplify:UpdateBranch
	amplify:UpdateDomainAssociation
	amplify:UpdateWebHook

Prefixo do serviço	Ações do
amplifyuibuilder	amplifyuibuilder:CreateComponent amplifyuibuilder:CreateForm amplifyuibuilder:CreateTheme amplifyuibuilder:DeleteComponent amplifyuibuilder:DeleteForm amplifyuibuilder:DeleteTheme amplifyuibuilder:ExportComponents amplifyuibuilder:ExportThemes amplifyuibuilder:GetCodegenJob amplifyuibuilder:ListCodegenJobs amplifyuibuilder:ListComponents amplifyuibuilder:ListForms amplifyuibuilder:ListThemes amplifyuibuilder:ResetMetadataFlag amplifyuibuilder:StartCodegenJob amplifyuibuilder:UpdateComponent amplifyuibuilder:UpdateForm amplifyuibuilder:UpdateTheme

Prefixo do serviço	Ações do
app-integrations	app-integrations:CreateApplication
	app-integrations:CreateDataIntegration
	app-integrations:CreateDataIntegrationAssociation
	app-integrations:CreateEventIntegration
	app-integrations:DeleteApplication
	app-integrations:DeleteDataIntegration
	app-integrations:DeleteEventIntegration
	app-integrations:GetApplication
	app-integrations:GetDataIntegration
	app-integrations:GetEventIntegration
	app-integrations:ListApplicationAssociations
	app-integrations:ListApplications
	app-integrations:ListDataIntegrationAssociations
	app-integrations:ListDataIntegrations
	app-integrations:ListEventIntegrationAssociations
	app-integrations:ListEventIntegrations
	app-integrations:UpdateApplication
	app-integrations:UpdateDataIntegration
	app-integrations:UpdateDataIntegrationAssociation
	app-integrations:UpdateEventIntegration

Prefixo do serviço	Ações do
appconfig	appconfig:CreateApplication appconfig:CreateConfigurationProfile appconfig:CreateDeploymentStrategy appconfig:CreateEnvironment appconfig:CreateExtension appconfig:CreateExtensionAssociation appconfig:CreateHostedConfigurationVersion appconfig:DeleteApplication appconfig:DeleteConfigurationProfile appconfig:DeleteDeploymentStrategy appconfig:DeleteEnvironment appconfig:DeleteExtension appconfig:DeleteExtensionAssociation appconfig:DeleteHostedConfigurationVersion appconfig:GetAccountSettings appconfig:GetApplication appconfig:GetConfiguration appconfig:GetConfigurationProfile appconfig:GetDeployment appconfig:GetDeploymentStrategy appconfig:GetEnvironment

Prefixo do serviço	Ações do
	appconfig:GetExtension appconfig:GetExtensionAssociation appconfig:GetHostedConfigurationVersion appconfig:ListApplications appconfig:ListConfigurationProfiles appconfig:ListDeploymentStrategies appconfig:ListDeployments appconfig:ListEnvironments appconfig:ListExtensionAssociations appconfig:ListExtensions appconfig:ListHostedConfigurationVersions appconfig:StartDeployment appconfig:StopDeployment appconfig:UpdateAccountSettings appconfig:UpdateApplication appconfig:UpdateConfigurationProfile appconfig:UpdateDeploymentStrategy appconfig:UpdateEnvironment appconfig:UpdateExtension appconfig:UpdateExtensionAssociation appconfig:ValidateConfiguration

Prefixo do serviço	Ações do
appflow	appflow:CancelFlowExecutions
	appflow:CreateConnectorProfile
	appflow:CreateFlow
	appflow:DeleteConnectorProfile
	appflow:DeleteFlow
	appflow:DescribeConnector
	appflow:DescribeConnectorEntity
	appflow:DescribeConnectorProfiles
	appflow:DescribeConnectors
	appflow:DescribeFlow
	appflow:DescribeFlowExecutionRecords
	appflow:ListConnectorEntities
	appflow:ListConnectors
	appflow:ListFlows
	appflow:RegisterConnector
	appflow:ResetConnectorMetadataCache
	appflow:StartFlow
	appflow:StopFlow
	appflow:UnRegisterConnector
	appflow:UpdateConnectorProfile
	appflow:UpdateConnectorRegistration

Prefixo do serviço	Ações do
	appflow:UpdateFlow

Prefixo do serviço	Ações do
applicationinsights	applicationinsights:AddWorkload applicationinsights:CreateApplication applicationinsights:CreateComponent applicationinsights:CreateLogPattern applicationinsights:DeleteApplication applicationinsights:DeleteComponent applicationinsights:DeleteLogPattern applicationinsights:DescribeApplication applicationinsights:DescribeComponent applicationinsights:DescribeComponentConfiguration applicationinsights:DescribeComponentConfigurationRecommendation applicationinsights:DescribeLogPattern applicationinsights:DescribeObservation applicationinsights:DescribeProblem applicationinsights:DescribeProblemObservations applicationinsights:DescribeWorkload applicationinsights:ListApplications applicationinsights:ListComponents applicationinsights:ListConfigurationHistory applicationinsights:ListLogPatternSets

Prefixo do serviço	Ações do
	applicationinsights:ListLogPatterns applicationinsights:ListProblems applicationinsights:ListWorkloads applicationinsights:RemoveWorkload applicationinsights:UpdateApplication applicationinsights:UpdateComponent applicationinsights:UpdateComponentConfiguration applicationinsights:UpdateLogPattern applicationinsights:UpdateWorkload

Prefixo do serviço	Ações do
appmesh	appmesh:CreateGatewayRoute
	appmesh:CreateMesh
	appmesh:CreateRoute
	appmesh:CreateVirtualGateway
	appmesh:CreateVirtualNode
	appmesh:CreateVirtualRouter
	appmesh:CreateVirtualService
	appmesh:DeleteGatewayRoute
	appmesh:DeleteMesh
	appmesh:DeleteRoute
	appmesh:DeleteVirtualGateway
	appmesh:DeleteVirtualNode
	appmesh:DeleteVirtualRouter
	appmesh:DeleteVirtualService
	appmesh:DescribeGatewayRoute
	appmesh:DescribeMesh
	appmesh:DescribeRoute
	appmesh:DescribeVirtualGateway
	appmesh:DescribeVirtualNode
	appmesh:DescribeVirtualRouter
	appmesh:DescribeVirtualService

Prefixo do serviço	Ações do
	appmesh:ListGatewayRoutes
	appmesh:ListMeshes
	appmesh:ListRoutes
	appmesh:ListVirtualGateways
	appmesh:ListVirtualNodes
	appmesh:ListVirtualRouters
	appmesh:ListVirtualServices
	appmesh:StreamAggregatedResources
	appmesh:UpdateGatewayRoute
	appmesh:UpdateMesh
	appmesh:UpdateRoute
	appmesh:UpdateVirtualGateway
	appmesh:UpdateVirtualNode
	appmesh:UpdateVirtualRouter
	appmesh:UpdateVirtualService

Prefixo do serviço	Ações do
appstream	appstream:AssociateAppBlockBuilderAppBlock appstream:AssociateApplicationFleet appstream:AssociateApplicationToEntitlement appstream:AssociateFleet appstream:AssociateSoftwareToImageBuilder appstream:BatchAssociateUserStack appstream:BatchDisassociateUserStack appstream:CopyImage appstream:CreateAppBlock appstream:CreateAppBlockBuilder appstream:CreateAppBlockBuilderStreamingURL appstream:CreateApplication appstream:CreateDirectoryConfig appstream:CreateEntitlement appstream:CreateFleet appstream:CreateImageBuilder appstream:CreateImageBuilderStreamingURL appstream:CreateStack appstream:CreateStreamingURL appstream:CreateThemeForStack appstream:CreateUpdatedImage

Prefixo do serviço	Ações do
	appstream:CreateUsageReportSubscription appstream:CreateUser appstream:DeleteAppBlock appstream:DeleteAppBlockBuilder appstream:DeleteApplication appstream:DeleteDirectoryConfig appstream:DeleteEntitlement appstream:DeleteFleet appstream:DeleteImage appstream:DeleteImageBuilder appstream:DeleteImagePermissions appstream:DeleteStack appstream:DeleteThemeForStack appstream:DeleteUsageReportSubscription appstream:DeleteUser appstream:DescribeAppBlockBuilderAppBlockAssociations appstream:DescribeAppBlockBuilders appstream:DescribeAppBlocks appstream:DescribeAppLicenseUsage appstream:DescribeApplicationFleetAssociations appstream:DescribeApplications

Prefixo do serviço	Ações do
	appstream:DescribeDirectoryConfigs appstream:DescribeEntitlements appstream:DescribeFleets appstream:DescribeImageBuilders appstream:DescribeImagePermissions appstream:DescribeImages appstream:DescribeSessions appstream:DescribeStacks appstream:DescribeThemeForStack appstream:DescribeUsageReportSubscriptions appstream:DescribeUserStackAssociations appstream:DescribeUsers appstream:DisableUser appstream:DisassociateAppBlockBuilderAppBlock appstream:DisassociateApplicationFleet appstream:DisassociateApplicationFromEntitlement appstream:DisassociateFleet appstream:DisassociateSoftwareFromImageBuilder appstream:EnableUser appstream:ExpireSession appstream:GetExportImageTask

Prefixo do serviço	Ações do
	appstream:ListAssociatedFleets appstream:ListAssociatedStacks appstream:ListEntitledApplications appstream:ListExportImageTasks appstream:StartAppBlockBuilder appstream:StartFleet appstream:StartImageBuilder appstream:StartSoftwareDeploymentToImageBuilder appstream:StopAppBlockBuilder appstream:StopFleet appstream:StopImageBuilder appstream:UpdateAppBlockBuilder appstream:UpdateApplication appstream:UpdateDirectoryConfig appstream:UpdateEntitlement appstream:UpdateFleet appstream:UpdateImagePermissions appstream:UpdateStack appstream:UpdateThemeForStack

Prefixo do serviço	Ações do
appsync	appsync:AssociateApi
	appsync:AssociateMergedGraphqlApi
	appsync:AssociateSourceGraphqlApi
	appsync:CreateApi
	appsync:CreateApiCache
	appsync:CreateApiKey
	appsync:CreateChannelNamespace
	appsync:CreateDataSource
	appsync:CreateDomainName
	appsync:CreateFunction
	appsync:CreateGraphqlApi
	appsync:CreateResolver
	appsync:CreateType
	appsync:DeleteApi
	appsync:DeleteApiCache
	appsync:DeleteApiKey
	appsync:DeleteChannelNamespace
	appsync:DeleteDataSource
	appsync:DeleteDomainName
	appsync:DeleteFunction
	appsync:DeleteGraphqlApi

Prefixo do serviço	Ações do
	appsync:DeleteResolver
	appsync:DeleteType
	appsync:DisassociateApi
	appsync:DisassociateMergedGraphqlApi
	appsync:DisassociateSourceGraphqlApi
	appsync:EvaluateCode
	appsync:EvaluateMappingTemplate
	appsync:FlushApiCache
	appsync:GetApi
	appsync:GetApiAssociation
	appsync:GetApiCache
	appsync:GetChannelNamespace
	appsync:GetDataSource
	appsync:GetDataSourceIntrospection
	appsync:GetDomainName
	appsync:GetFunction
	appsync:GetGraphqlApi
	appsync:GetGraphqlApiEnvironmentVariables
	appsync:GetIntrospectionSchema
	appsync:GetResolver
	appsync:GetSchemaCreationStatus

Prefixo do serviço	Ações do
	appsync:GetSourceApiAssociation
	appsync:GetType
	appsync:ListApiKeys
	appsync:ListApis
	appsync:ListChannelNamespaces
	appsync:ListDataSources
	appsync:ListDomainNames
	appsync:ListFunctions
	appsync:ListGraphqlApis
	appsync:ListResolvers
	appsync:ListResolversByFunction
	appsync:ListSourceApiAssociations
	appsync:ListTypes
	appsync:ListTypesByAssociation
	appsync:PutGraphqlApiEnvironmentVariables
	appsync:StartDataSourceIntrospection
	appsync:StartSchemaCreation
	appsync:StartSchemaMerge
	appsync:UpdateApi
	appsync:UpdateApiCache
	appsync:UpdateApiKey

Prefixo do serviço	Ações do
	appsync:UpdateChannelNamespace
	appsync:UpdateDataSource
	appsync:UpdateDomainName
	appsync:UpdateFunction
	appsync:UpdateGraphqlApi
	appsync:UpdateResolver
	appsync:UpdateSourceApiAssociation
	appsync:UpdateType

Prefixo do serviço	Ações do
aps	aps:CreateAlertManagerDefinition aps:CreateAnomalyDetector aps:CreateLoggingConfiguration aps:CreateQueryLoggingConfiguration aps:CreateRuleGroupsNamespace aps:CreateWorkspace aps:DeleteAlertManagerDefinition aps:DeleteAnomalyDetector aps:DeleteLoggingConfiguration aps:DeleteQueryLoggingConfiguration aps:DeleteResourcePolicy aps:DeleteRuleGroupsNamespace aps:DeleteScraper aps:DeleteScraperLoggingConfiguration aps:DeleteWorkspace aps:DescribeAlertManagerDefinition aps:DescribeAnomalyDetector aps:DescribeLoggingConfiguration aps:DescribeQueryLoggingConfiguration aps:DescribeResourcePolicy aps:DescribeRuleGroupsNamespace

Prefixo do serviço	Ações do
	aps:DescribeScraper aps:DescribeScraperLoggingConfiguration aps:DescribeWorkspace aps:DescribeWorkspaceConfiguration aps:GetDefaultScraperConfiguration aps:ListAnomalyDetectors aps:ListRuleGroupsNamespaces aps:ListScrapers aps:ListWorkspaces aps:PutAlertManagerDefinition aps:PutAnomalyDetector aps:PutResourcePolicy aps:PutRuleGroupsNamespace aps:UpdateLoggingConfiguration aps:UpdateQueryLoggingConfiguration aps:UpdateScraper aps:UpdateScraperLoggingConfiguration aps:UpdateWorkspaceAlias aps:UpdateWorkspaceConfiguration

Prefixo do serviço	Ações do
athena	athena:BatchGetNamedQuery
	athena:BatchGetPreparedStatement
	athena:BatchGetQueryExecution
	athena:CancelCapacityReservation
	athena:CreateCapacityReservation
	athena:CreateDataCatalog
	athena:CreateNamedQuery
	athena:CreateNotebook
	athena:CreatePreparedStatement
	athena:CreatePresignedNotebookUrl
	athena:CreateWorkGroup
	athena:DeleteCapacityReservation
	athena:DeleteDataCatalog
	athena:DeleteNamedQuery
	athena:DeleteNotebook
	athena:DeletePreparedStatement
	athena:DeleteWorkGroup
	athena:ExportNotebook
	athena:GetCalculationExecution
	athena:GetCalculationExecutionCode
	athena:GetCalculationExecutionStatus

Prefixo do serviço	Ações do
	athena:GetCapacityAssignmentConfiguration
	athena:GetCapacityReservation
	athena:GetDataCatalog
	athena:GetDatabase
	athena:GetNamedQuery
	athena:GetNotebookMetadata
	athena:GetPreparedStatement
	athena:GetQueryExecution
	athena:GetQueryResults
	athena:GetQueryResultsStream
	athena:GetQueryRuntimeStatistics
	athena:GetResourceDashboard
	athena:GetSession
	athena:GetSessionEndpoint
	athena:GetSessionStatus
	athena:GetTableMetadata
	athena:GetWorkGroup
	athena:ImportNotebook
	athena:ListApplicationDPUSizes
	athena:ListCalculationExecutions
	athena:ListCapacityReservations

Prefixo do serviço	Ações do
	athena:ListDataCatalogs
	athena:ListDatabases
	athena:ListEngineVersions
	athena:ListExecutors
	athena:ListNamedQueries
	athena:ListNotebookMetadata
	athena:ListNotebookSessions
	athena:ListPreparedStatements
	athena:ListQueryExecutions
	athena:ListSessions
	athena:ListTableMetadata
	athena:ListWorkGroups
	athena:PutCapacityAssignmentConfiguration
	athena:StartCalculationExecution
	athena:StartQueryExecution
	athena:StartSession
	athena:StopCalculationExecution
	athena:StopQueryExecution
	athena:TerminateSession
	athena:UpdateCapacityReservation
	athena:UpdateDataCatalog

Prefixo do serviço	Ações do
	athena:UpdateNamedQuery
	athena:UpdateNotebook
	athena:UpdateNotebookMetadata
	athena:UpdatePreparedStatement
	athena:UpdateWorkGroup

Prefixo do serviço	Ações do
auditmanager	auditmanager:AssociateAssessmentReportEvidenceFolder auditmanager:BatchAssociateAssessmentReportEvidence auditmanager:BatchCreateDelegationByAssessment auditmanager:BatchDeleteDelegationByAssessment auditmanager:BatchDisassociateAssessmentReportEvidence auditmanager:BatchImportEvidenceToAssessmentControl auditmanager:CreateAssessment auditmanager:CreateAssessmentFramework auditmanager:CreateAssessmentReport auditmanager:CreateControl auditmanager:DeleteAssessment auditmanager:DeleteAssessmentFramework auditmanager:DeleteAssessmentFrameworkShare auditmanager:DeleteAssessmentReport auditmanager:DeleteControl auditmanager:DeregisterAccount auditmanager:DeregisterOrganizationAdminAccount auditmanager:DisassociateAssessmentReportEvidenceFolder auditmanager:GetAccountStatus auditmanager:GetAssessment auditmanager:GetAssessmentFramework

Prefixo do serviço	Ações do
	auditmanager:GetAssessmentReportUrl auditmanager:GetChangeLogs auditmanager:GetControl auditmanager:GetDelegations auditmanager:GetEvidence auditmanager:GetEvidenceByEvidenceFolder auditmanager:GetEvidenceFileUploadUrl auditmanager:GetEvidenceFolder auditmanager:GetEvidenceFoldersByAssessment auditmanager:GetEvidenceFoldersByAssessmentControl auditmanager:GetInsights auditmanager:GetInsightsByAssessment auditmanager:GetOrganizationAdminAccount auditmanager:GetServicesInScope auditmanager:GetSettings auditmanager:ListAssessmentControlInsightsByControlDomain auditmanager:ListAssessmentFrameworkShareRequests auditmanager:ListAssessmentFrameworks auditmanager:ListAssessmentReports auditmanager:ListAssessments auditmanager:ListControlDomainInsights

Prefixo do serviço	Ações do
	auditmanager:ListControlDomainInsightsByAssessment auditmanager:ListControlInsightsByControlDomain auditmanager:ListControls auditmanager:ListKeywordsForDataSource auditmanager:ListNotifications auditmanager:RegisterAccount auditmanager:RegisterOrganizationAdminAccount auditmanager:StartAssessmentFrameworkShare auditmanager:UpdateAssessment auditmanager:UpdateAssessmentControl auditmanager:UpdateAssessmentControlSetStatus auditmanager:UpdateAssessmentFramework auditmanager:UpdateAssessmentFrameworkShare auditmanager:UpdateAssessmentStatus auditmanager:UpdateControl auditmanager:UpdateSettings auditmanager:ValidateAssessmentReportIntegrity

Prefixo do serviço	Ações do
ajuste de escala automático	autoscaling:AttachInstances
	autoscaling:AttachLoadBalancerTargetGroups
	autoscaling:AttachLoadBalancers
	autoscaling:AttachTrafficSources
	autoscaling:BatchDeleteScheduledAction
	autoscaling:BatchPutScheduledUpdateGroupAction
	autoscaling:CancelInstanceRefresh
	autoscaling:CompleteLifecycleAction
	autoscaling:CreateAutoScalingGroup
	autoscaling:CreateLaunchConfiguration
	autoscaling:DeleteAutoScalingGroup
	autoscaling:DeleteLaunchConfiguration
	autoscaling:DeleteLifecycleHook
	autoscaling:DeleteNotificationConfiguration
	autoscaling:DeletePolicy
	autoscaling:DeleteScheduledAction
	autoscaling:DeleteWarmPool
	autoscaling:DescribeAccountLimits
	autoscaling:DescribeAdjustmentTypes
	autoscaling:DescribeAutoScalingGroups
	autoscaling:DescribeAutoScalingInstances

Prefixo do serviço	Ações do
	autoscaling:DescribeAutoScalingNotificationTypes
	autoscaling:DescribeInstanceRefreshes
	autoscaling:DescribeLaunchConfigurations
	autoscaling:DescribeLifecycleHookTypes
	autoscaling:DescribeLifecycleHooks
	autoscaling:DescribeLoadBalancerTargetGroups
	autoscaling:DescribeLoadBalancers
	autoscaling:DescribeMetricCollectionTypes
	autoscaling:DescribeNotificationConfigurations
	autoscaling:DescribePolicies
	autoscaling:DescribeScalingActivities
	autoscaling:DescribeScalingProcessTypes
	autoscaling:DescribeScheduledActions
	autoscaling:DescribeTerminationPolicyTypes
	autoscaling:DescribeTrafficSources
	autoscaling:DescribeWarmPool
	autoscaling:DetachInstances
	autoscaling:DetachLoadBalancerTargetGroups
	autoscaling:DetachLoadBalancers
	autoscaling:DetachTrafficSources
	autoscaling:DisableMetricsCollection

Prefixo do serviço	Ações do
	autoscaling:EnableMetricsCollection autoscaling:EnterStandby autoscaling:ExecutePolicy autoscaling:ExitStandby autoscaling:GetPredictiveScalingForecast autoscaling:PutLifecycleHook autoscaling:PutNotificationConfiguration autoscaling:PutScalingPolicy autoscaling:PutScheduledUpdateGroupAction autoscaling:PutWarmPool autoscaling:RecordLifecycleActionHeartbeat autoscaling:ResumeProcesses autoscaling:RollbackInstanceRefresh autoscaling:SetDesiredCapacity autoscaling:SetInstanceHealth autoscaling:SetInstanceProtection autoscaling:StartInstanceRefresh autoscaling:SuspendProcesses autoscaling:TerminateInstanceInAutoScalingGroup autoscaling:UpdateAutoScalingGroup
aws-marketplace	aws-marketplace:GetEntitlements

Prefixo do serviço	Ações do
backup	backup:CancelLegalHold
	backup:CreateBackupPlan
	backup:CreateBackupSelection
	backup:CreateBackupVault
	backup:CreateFramework
	backup:CreateLegalHold
	backup:CreateLogicallyAirGappedBackupVault
	backup:CreateReportPlan
	backup:CreateRestoreAccessBackupVault
	backup:CreateRestoreTestingPlan
	backup:CreateRestoreTestingSelection
	backup:CreateTieringConfiguration
	backup:DeleteBackupPlan
	backup:DeleteBackupSelection
	backup:DeleteBackupVault
	backup:DeleteBackupVaultAccessPolicy
	backup:DeleteBackupVaultLockConfiguration
	backup:DeleteBackupVaultNotifications
	backup:DeleteFramework
	backup:DeleteRecoveryPoint
	backup:DeleteReportPlan

Prefixo do serviço	Ações do
	backup:DeleteRestoreTestingPlan
	backup:DeleteRestoreTestingSelection
	backup:DeleteTieringConfiguration
	backup:DescribeBackupJob
	backup:DescribeBackupVault
	backup:DescribeCopyJob
	backup:DescribeFramework
	backup:DescribeGlobalSettings
	backup:DescribeProtectedResource
	backup:DescribeRecoveryPoint
	backup:DescribeRegionSettings
	backup:DescribeReportJob
	backup:DescribeReportPlan
	backup:DescribeRestoreJob
	backup:DescribeScanJob
	backup:DisassociateRecoveryPoint
	backup:DisassociateRecoveryPointFromParent
	backup:ExportBackupPlanTemplate
	backup:GetBackupPlan
	backup:GetBackupPlanFromJSON
	backup:GetBackupPlanFromTemplate

Prefixo do serviço	Ações do
	backup:GetBackupSelection
	backup:GetBackupVaultAccessPolicy
	backup:GetBackupVaultNotifications
	backup:GetLegalHold
	backup:GetRecoveryPointRestoreMetadata
	backup:GetRestoreJobMetadata
	backup:GetRestoreTestingInferredMetadata
	backup:GetRestoreTestingPlan
	backup:GetRestoreTestingSelection
	backup:GetSupportedResourceTypes
	backup:GetTieringConfiguration
	backup:ListBackupJobSummaries
	backup:ListBackupJobs
	backup:ListBackupPlanTemplates
	backup:ListBackupPlanVersions
	backup:ListBackupPlans
	backup:ListBackupSelections
	backup:ListBackupVaults
	backup:ListCopyJobSummaries
	backup:ListCopyJobs
	backup:ListFrameworks

Prefixo do serviço	Ações do
	backup:ListIndexedRecoveryPoints
	backup:ListLegalHolds
	backup:ListProtectedResources
	backup:ListRecoveryPointsByBackupVault
	backup:ListRecoveryPointsByLegalHold
	backup:ListRecoveryPointsByResource
	backup:ListReportJobs
	backup:ListReportPlans
	backup:ListRestoreAccessBackupVaults
	backup:ListRestoreJobSummaries
	backup:ListRestoreJobs
	backup:ListRestoreJobsByProtectedResource
	backup:ListRestoreTestingPlans
	backup:ListRestoreTestingSelections
	backup:ListScanJobSummaries
	backup:ListScanJobs
	backup:ListTieringConfigurations
	backup:PutBackupVaultAccessPolicy
	backup:PutBackupVaultLockConfiguration
	backup:PutBackupVaultNotifications
	backup:PutRestoreValidationResult

Prefixo do serviço	Ações do
	backup:StartBackupJob
	backup:StartCopyJob
	backup:StartReportJob
	backup:StartRestoreJob
	backup:StopBackupJob
	backup:UpdateBackupPlan
	backup:UpdateFramework
	backup:UpdateGlobalSettings
	backup:UpdateRecoveryPointLifecycle
	backup:UpdateRegionSettings
	backup:UpdateReportPlan
	backup:UpdateRestoreTestingPlan
	backup:UpdateRestoreTestingSelection
	backup:UpdateTieringConfiguration

Prefixo do serviço	Ações do
lote	batch:CancelJob batch:CreateComputeEnvironment batch:CreateConsumableResource batch:CreateJobQueue batch:CreateQuotaShare batch:CreateSchedulingPolicy batch:CreateServiceEnvironment batch:DeleteComputeEnvironment batch:DeleteConsumableResource batch:DeleteJobQueue batch:DeleteQuotaShare batch:DeleteSchedulingPolicy batch:DeleteServiceEnvironment batch:DeregisterJobDefinition batch:DescribeComputeEnvironments batch:DescribeConsumableResource batch:DescribeJobDefinitions batch:DescribeJobQueues batch:DescribeJobs batch:DescribeQuotaShare batch:DescribeSchedulingPolicies

Prefixo do serviço	Ações do
	batch:DescribeServiceEnvironments batch:DescribeServiceJob batch:GetJobQueueSnapshot batch:ListConsumableResources batch:ListJobs batch:ListJobsByConsumableResource batch:ListQuotaShares batch:ListSchedulingPolicies batch:ListServiceJobs batch:RegisterJobDefinition batch:SubmitJob batch:SubmitServiceJob batch:TerminateJob batch:TerminateServiceJob batch:UpdateComputeEnvironment batch:UpdateConsumableResource batch:UpdateJobQueue batch:UpdateQuotaShare batch:UpdateSchedulingPolicy batch:UpdateServiceEnvironment batch:UpdateServiceJob

Prefixo do serviço	Ações do
braket	braket:CancelJob
	braket:CancelQuantumTask
	braket:CreateJob
	braket:CreateQuantumTask
	braket:CreateSpendingLimit
	braket:GetDevice
	braket:GetJob
	braket:GetQuantumTask
	braket:SearchDevices
	braket:SearchJobs
	braket:SearchQuantumTasks
	braket:SearchSpendingLimits

Prefixo do serviço	Ações do
orçamentos	budgets:CreateBudgetAction budgets:DeleteBudgetAction budgets:DescribeBudgetAction budgets:DescribeBudgetActionHistories budgets:DescribeBudgetActionsForAccount budgets:DescribeBudgetActionsForBudget budgets:ExecuteBudgetAction budgets:ModifyBudget budgets:UpdateBudgetAction budgets:ViewBudget
cloud9	cloud9:CreateEnvironmentEC2 cloud9:CreateEnvironmentMembership cloud9:DeleteEnvironment cloud9:DeleteEnvironmentMembership cloud9:DescribeEnvironmentMemberships cloud9:DescribeEnvironmentStatus cloud9:DescribeEnvironments cloud9:ListEnvironments cloud9:UpdateEnvironment cloud9:UpdateEnvironmentMembership

Prefixo do serviço	Ações do
cloudformation	cloudformation:BatchDescribeTypeConfigurations cloudformation:CancelUpdateStack cloudformation:ContinueUpdateRollback cloudformation:CreateChangeSet cloudformation:CreateGeneratedTemplate cloudformation:CreateStack cloudformation:CreateStackInstances cloudformation:CreateStackSet cloudformation:DeactivateType cloudformation:DeleteChangeSet cloudformation:DeleteGeneratedTemplate cloudformation:DeleteStack cloudformation:DeleteStackInstances cloudformation:DeleteStackSet cloudformation:DeregisterType cloudformation:DescribeAccountLimits cloudformation:DescribeChangeSet cloudformation:DescribeChangeSetHooks cloudformation:DescribeEvents cloudformation:DescribeGeneratedTemplate cloudformation:DescribeOrganizationsAccess

Prefixo do serviço	Ações do
	cloudformation:DescribePublisher cloudformation:DescribeResourceScan cloudformation:DescribeStackDriftDetectionStatus cloudformation:DescribeStackEvents cloudformation:DescribeStackInstance cloudformation:DescribeStackResource cloudformation:DescribeStackResourceDrifts cloudformation:DescribeStackResources cloudformation:DescribeStackSet cloudformation:DescribeStackSetOperation cloudformation:DescribeStacks cloudformation:DescribeType cloudformation:DescribeTypeRegistration cloudformation:DetectStackDrift cloudformation:DetectStackResourceDrift cloudformation:DetectStackSetDrift cloudformation:EstimateTemplateCost cloudformation:ExecuteChangeSet cloudformation:GetGeneratedTemplate cloudformation:GetHookResult cloudformation:GetStackPolicy

Prefixo do serviço	Ações do
	cloudformation:GetTemplate cloudformation:GetTemplateSummary cloudformation:ImportStacksToStackSet cloudformation:ListChangeSets cloudformation:ListExports cloudformation:ListGeneratedTemplates cloudformation:ListHookResults cloudformation:ListImports cloudformation:ListResourceScanRelatedResources cloudformation:ListResourceScanResources cloudformation:ListResourceScans cloudformation:ListStackInstanceResourceDrifts cloudformation:ListStackInstances cloudformation:ListStackRefactors cloudformation:ListStackResources cloudformation:ListStackSetAutoDeploymentTargets cloudformation:ListStackSetOperationResults cloudformation:ListStackSetOperations cloudformation:ListStackSets cloudformation:ListTypeRegistrations cloudformation:ListTypeVersions

Prefixo do serviço	Ações do
	cloudformation:ListTypes cloudformation:PublishType cloudformation:RecordHandlerProgress cloudformation:RegisterPublisher cloudformation:RegisterType cloudformation:RollbackStack cloudformation:SetStackPolicy cloudformation:SetTypeConfiguration cloudformation:SetTypeDefaultVersion cloudformation:SignalResource cloudformation:StartResourceScan cloudformation:StopStackSetOperation cloudformation:TestType cloudformation:UpdateGeneratedTemplate cloudformation:UpdateStack cloudformation:UpdateStackInstances cloudformation:UpdateStackSet cloudformation:UpdateTerminationProtection cloudformation:ValidateTemplate

Prefixo do serviço	Ações do
cloudfront	cloudfront:AssociateAlias
	cloudfront:AssociateDistributionTenantWebACL
	cloudfront:AssociateDistributionWebACL
	cloudfront:CreateCachePolicy
	cloudfront:CreateCloudFrontOriginAccessIdentity
	cloudfront:CreateConnectionFunction
	cloudfront:CreateContinuousDeploymentPolicy
	cloudfront:CreateDistributionTenant
	cloudfront:CreateFieldLevelEncryptionConfig
	cloudfront:CreateFieldLevelEncryptionProfile
	cloudfront:CreateFunction
	cloudfront:CreateInvalidation
	cloudfront:CreateKeyGroup
	cloudfront:CreateKeyValueStore
	cloudfront:CreateMonitoringSubscription
	cloudfront:CreateOriginAccessControl
	cloudfront:CreateOriginRequestPolicy
	cloudfront:CreatePublicKey
	cloudfront:CreateRealtimeLogConfig
	cloudfront:CreateResponseHeadersPolicy
	cloudfront:CreateTrustStore

Prefixo do serviço	Ações do
	cloudfront:DeleteAnycastIpList
	cloudfront:DeleteCachePolicy
	cloudfront:DeleteCloudFrontOriginAccessIdentity
	cloudfront:DeleteConnectionFunction
	cloudfront:DeleteConnectionGroup
	cloudfront:DeleteContinuousDeploymentPolicy
	cloudfront:DeleteDistribution
	cloudfront:DeleteDistributionTenant
	cloudfront:DeleteFieldLevelEncryptionConfig
	cloudfront:DeleteFieldLevelEncryptionProfile
	cloudfront:DeleteFunction
	cloudfront:DeleteKeyGroup
	cloudfront:DeleteKeyValueStore
	cloudfront:DeleteMonitoringSubscription
	cloudfront:DeleteOriginAccessControl
	cloudfront:DeleteOriginRequestPolicy
	cloudfront:DeletePublicKey
	cloudfront:DeleteRealtimeLogConfig
	cloudfront:DeleteResponseHeadersPolicy
	cloudfront:DeleteStreamingDistribution
	cloudfront:DeleteTrustStore

Prefixo do serviço	Ações do
	cloudfront:DeleteVpcOrigin
	cloudfront:DescribeFunction
	cloudfront:DescribeKeyValueStore
	cloudfront:DisassociateDistributionTenantWebACL
	cloudfront:DisassociateDistributionWebACL
	cloudfront:GetAnycastIpList
	cloudfront:GetCachePolicy
	cloudfront:GetCachePolicyConfig
	cloudfront:GetCloudFrontOriginAccessIdentity
	cloudfront:GetCloudFrontOriginAccessIdentityConfig
	cloudfront:GetContinuousDeploymentPolicy
	cloudfront:GetContinuousDeploymentPolicyConfig
	cloudfront:GetDistributionConfig
	cloudfront:GetFieldLevelEncryption
	cloudfront:GetFieldLevelEncryptionConfig
	cloudfront:GetFieldLevelEncryptionProfile
	cloudfront:GetFieldLevelEncryptionProfileConfig
	cloudfront:GetFunction
	cloudfront:GetInvalidation
	cloudfront:GetInvalidationForDistributionTenant
	cloudfront:GetKeyGroup

Prefixo do serviço	Ações do
	cloudfront:GetKeyGroupConfig
	cloudfront:GetMonitoringSubscription
	cloudfront:GetOriginAccessControl
	cloudfront:GetOriginAccessControlConfig
	cloudfront:GetOriginRequestPolicy
	cloudfront:GetOriginRequestPolicyConfig
	cloudfront:GetPublicKey
	cloudfront:GetPublicKeyConfig
	cloudfront:GetRealtimeLogConfig
	cloudfront:GetResponseHeadersPolicy
	cloudfront:GetResponseHeadersPolicyConfig
	cloudfront:GetStreamingDistribution
	cloudfront:GetStreamingDistributionConfig
	cloudfront:GetVpcOrigin
	cloudfront:ListAnycastIpLists
	cloudfront:ListCachePolicies
	cloudfront:ListCloudFrontOriginAccessIdentities
	cloudfront:ListConflictingAliases
	cloudfront:ListConnectionFunctions
	cloudfront:ListConnectionGroups
	cloudfront:ListContinuousDeploymentPolicies

Prefixo do serviço	Ações do
	cloudfront:ListDistributionTenants
	cloudfront:ListDistributionTenantsByCustomization
	cloudfront:ListDistributions
	cloudfront:ListDistributionsByAnycastIpListId
	cloudfront:ListDistributionsByCachePolicyId
	cloudfront:ListDistributionsByConnectionMode
	cloudfront:ListDistributionsByKeyGroup
	cloudfront:ListDistributionsByOriginRequestPolicyId
	cloudfront:ListDistributionsByRealtimeLogConfig
	cloudfront:ListDistributionsByResponseHeadersPolicyId
	cloudfront:ListDistributionsByVpcOriginId
	cloudfront:ListDistributionsByWebACLId
	cloudfront:ListFieldLevelEncryptionConfigs
	cloudfront:ListFieldLevelEncryptionProfiles
	cloudfront:ListFunctions
	cloudfront:ListInvalidations
	cloudfront:ListInvalidationsForDistributionTenant
	cloudfront:ListKeyGroups
	cloudfront:ListKeyValueStores
	cloudfront:ListOriginAccessControls
	cloudfront:ListOriginRequestPolicies

Prefixo do serviço	Ações do
	cloudfront:ListPublicKeys
	cloudfront:ListRealtimeLogConfigs
	cloudfront:ListResponseHeadersPolicies
	cloudfront:ListStreamingDistributions
	cloudfront:ListTrustStores
	cloudfront:PublishConnectionFunction
	cloudfront:PublishFunction
	cloudfront:TestConnectionFunction
	cloudfront:TestFunction
	cloudfront:UpdateAnycastIpList
	cloudfront:UpdateCachePolicy
	cloudfront:UpdateCloudFrontOriginAccessIdentity
	cloudfront:UpdateConnectionFunction
	cloudfront:UpdateConnectionGroup
	cloudfront:UpdateContinuousDeploymentPolicy
	cloudfront:UpdateDistribution
	cloudfront:UpdateDistributionTenant
	cloudfront:UpdateFieldLevelEncryptionConfig
	cloudfront:UpdateFieldLevelEncryptionProfile
	cloudfront:UpdateFunction
	cloudfront:UpdateKeyGroup

Prefixo do serviço	Ações do
	cloudfront:UpdateKeyValueStore cloudfront:UpdateOriginAccessControl cloudfront:UpdateOriginRequestPolicy cloudfront:UpdatePublicKey cloudfront:UpdateRealtimeLogConfig cloudfront:UpdateResponseHeadersPolicy cloudfront:UpdateTrustStore
cloudhsm	cloudhsm:CreateHsm cloudhsm:DeleteBackup cloudhsm:DeleteHsm cloudhsm:DeleteResourcePolicy cloudhsm:DescribeBackups cloudhsm:DescribeClusters cloudhsm:GetResourcePolicy cloudhsm:InitializeCluster cloudhsm:ModifyBackupAttributes cloudhsm:ModifyCluster cloudhsm:PutResourcePolicy cloudhsm:RestoreBackup

Prefixo do serviço	Ações do
cloudsearch	cloudsearch:BuildSuggesters
	cloudsearch:CreateDomain
	cloudsearch:DefineAnalysisScheme
	cloudsearch:DefineExpression
	cloudsearch:DefineIndexField
	cloudsearch:DefineSuggester
	cloudsearch:DeleteAnalysisScheme
	cloudsearch:DeleteDomain
	cloudsearch:DeleteExpression
	cloudsearch:DeleteIndexField
	cloudsearch:DeleteSuggester
	cloudsearch:DescribeAnalysisSchemes
	cloudsearch:DescribeAvailabilityOptions
	cloudsearch:DescribeDomainEndpointOptions
	cloudsearch:DescribeDomains
	cloudsearch:DescribeExpressions
	cloudsearch:DescribeIndexFields
	cloudsearch:DescribeScalingParameters
	cloudsearch:DescribeServiceAccessPolicies
	cloudsearch:DescribeSuggesters
	cloudsearch:IndexDocuments

Prefixo do serviço	Ações do
	cloudsearch:ListDomainNames
	cloudsearch:UpdateAvailabilityOptions
	cloudsearch:UpdateDomainEndpointOptions
	cloudsearch:UpdateScalingParameters
	cloudsearch:UpdateServiceAccessPolicies

Prefixo do serviço	Ações do
cloudtrail	cloudtrail:CancelQuery cloudtrail:CreateChannel cloudtrail:CreateDashboard cloudtrail:CreateEventDataStore cloudtrail:CreateTrail cloudtrail:DeleteChannel cloudtrail:DeleteDashboard cloudtrail:DeleteEventDataStore cloudtrail:DeleteResourcePolicy cloudtrail:DeleteTrail cloudtrail:DeregisterOrganizationDelegatedAdmin cloudtrail:DescribeQuery cloudtrail:DescribeTrails cloudtrail:DisableFederation cloudtrail:GenerateQuery cloudtrail:GetChannel cloudtrail:GetDashboard cloudtrail:GetEventConfiguration cloudtrail:GetEventDataStore cloudtrail:GetEventDataStoreData cloudtrail:GetEventSelectors

Prefixo do serviço	Ações do
	cloudtrail:GetImport cloudtrail:GetInsightSelectors cloudtrail:GetResourcePolicy cloudtrail:GetTrail cloudtrail:GetTrailStatus cloudtrail:ListChannels cloudtrail:ListDashboards cloudtrail:ListEventDataStores cloudtrail:ListImportFailures cloudtrail:ListImports cloudtrail:ListInsightsData cloudtrail:ListPublicKeys cloudtrail:ListQueries cloudtrail:ListTrails cloudtrail:LookupEvents cloudtrail:PutEventConfiguration cloudtrail:PutEventSelectors cloudtrail:PutInsightSelectors cloudtrail:PutResourcePolicy cloudtrail:RegisterOrganizationDelegatedAdmin cloudtrail:RestoreEventDataStore

Prefixo do serviço	Ações do
	cloudtrail:SearchSampleQueries cloudtrail:StartEventDataStoreIngestion cloudtrail:StartImport cloudtrail:StartLogging cloudtrail:StartQuery cloudtrail:StopEventDataStoreIngestion cloudtrail:StopImport cloudtrail:StopLogging cloudtrail:UpdateChannel cloudtrail:UpdateDashboard cloudtrail:UpdateEventDataStore cloudtrail:UpdateTrail

Prefixo do serviço	Ações do
cloudwatch	cloudwatch:DeleteAlarmMuteRule
	cloudwatch:DeleteAlarms
	cloudwatch:DeleteAnomalyDetector
	cloudwatch:DeleteDashboards
	cloudwatch:DeleteInsightRules
	cloudwatch:DeleteMetricStream
	cloudwatch:DescribeAlarmHistory
	cloudwatch:DescribeAlarms
	cloudwatch:DescribeAlarmsForMetric
	cloudwatch:DescribeAnomalyDetectors
	cloudwatch:DescribeInsightRules
	cloudwatch:DisableAlarmActions
	cloudwatch:DisableInsightRules
	cloudwatch:EnableAlarmActions
	cloudwatch:EnableInsightRules
	cloudwatch:GetAlarmMuteRule
	cloudwatch:GetDashboard
	cloudwatch:GetDataset
	cloudwatch:GetInsightRuleReport
	cloudwatch:GetMetricStatistics
	cloudwatch:GetMetricStream

Prefixo do serviço	Ações do
	cloudwatch:ListAlarmMuteRules
	cloudwatch:ListDashboards
	cloudwatch:ListManagedInsightRules
	cloudwatch:ListMetricStreams
	cloudwatch:PutAlarmMuteRule
	cloudwatch:PutAnomalyDetector
	cloudwatch:PutCompositeAlarm
	cloudwatch:PutDashboard
	cloudwatch:PutInsightRule
	cloudwatch:PutLogAlarm
	cloudwatch:PutManagedInsightRules
	cloudwatch:PutMetricAlarm
	cloudwatch:PutMetricStream
	cloudwatch:SetAlarmState
	cloudwatch:StartMetricStreams
	cloudwatch:StopMetricStreams

Prefixo do serviço	Ações do
codeartifact	codeartifact:AssociateExternalConnection codeartifact:CopyPackageVersions codeartifact:CreateDomain codeartifact:CreateRepository codeartifact:DeleteDomain codeartifact:DeleteDomainPermissionsPolicy codeartifact:DeletePackage codeartifact:DeletePackageVersions codeartifact:DeleteRepository codeartifact:DeleteRepositoryPermissionsPolicy codeartifact:DescribeDomain codeartifact:DescribePackage codeartifact:DescribePackageVersion codeartifact:DescribeRepository codeartifact:DisassociateExternalConnection codeartifact:DisposePackageVersions codeartifact:GetAssociatedPackageGroup codeartifact:GetAuthorizationToken codeartifact:GetDomainPermissionsPolicy codeartifact:GetPackageVersionAsset codeartifact:GetPackageVersionReadme

Prefixo do serviço	Ações do
	codeartifact:GetRepositoryEndpoint codeartifact:GetRepositoryPermissionsPolicy codeartifact:ListDomains codeartifact:ListPackageGroups codeartifact:ListPackageVersionAssets codeartifact:ListPackageVersionDependencies codeartifact:ListPackageVersions codeartifact:ListPackages codeartifact:ListRepositories codeartifact:ListRepositoriesInDomain codeartifact:PublishPackageVersion codeartifact:PutDomainPermissionsPolicy codeartifact:PutPackageMetadata codeartifact:PutPackageOriginConfiguration codeartifact:PutRepositoryPermissionsPolicy codeartifact:ReadFromRepository codeartifact:UpdatePackageVersionsStatus codeartifact:UpdateRepository

Prefixo do serviço	Ações do
codedeploy	codedeploy:BatchGetApplicationRevisions
	codedeploy:BatchGetApplications
	codedeploy:BatchGetDeploymentGroups
	codedeploy:BatchGetDeploymentInstances
	codedeploy:BatchGetDeploymentTargets
	codedeploy:BatchGetDeployments
	codedeploy:BatchGetOnPremisesInstances
	codedeploy:ContinueDeployment
	codedeploy:CreateApplication
	codedeploy:CreateDeployment
	codedeploy:CreateDeploymentConfig
	codedeploy:CreateDeploymentGroup
	codedeploy:DeleteApplication
	codedeploy:DeleteDeploymentConfig
	codedeploy:DeleteDeploymentGroup
	codedeploy:DeleteGitHubAccountToken
	codedeploy:DeleteResourcesByExternalId
	codedeploy:DeregisterOnPremisesInstance
	codedeploy:GetApplication
	codedeploy:GetApplicationRevision
	codedeploy:GetDeployment

Prefixo do serviço	Ações do
	codedeploy:GetDeploymentConfig
	codedeploy:GetDeploymentGroup
	codedeploy:GetDeploymentInstance
	codedeploy:GetDeploymentTarget
	codedeploy:GetOnPremisesInstance
	codedeploy:ListApplicationRevisions
	codedeploy:ListApplications
	codedeploy:ListDeploymentConfigs
	codedeploy:ListDeploymentGroups
	codedeploy:ListDeploymentInstances
	codedeploy:ListDeploymentTargets
	codedeploy:ListDeployments
	codedeploy:ListGitHubAccountTokenNames
	codedeploy:ListOnPremisesInstances
	codedeploy:PutLifecycleEventHookExecutionStatus
	codedeploy:RegisterApplicationRevision
	codedeploy:RegisterOnPremisesInstance
	codedeploy:SkipWaitTimeForInstanceTermination
	codedeploy:StopDeployment
	codedeploy:UpdateApplication
	codedeploy:UpdateDeploymentGroup

Prefixo do serviço	Ações do
codeguru-profiler	codeguru-profiler:AddNotificationChannels codeguru-profiler:BatchGetFrameMetricData codeguru-profiler:CreateProfilingGroup codeguru-profiler:DeleteProfilingGroup codeguru-profiler:DescribeProfilingGroup codeguru-profiler:GetFindingsReportAccountSummary codeguru-profiler:GetNotificationConfiguration codeguru-profiler:GetPolicy codeguru-profiler:GetProfile codeguru-profiler:GetRecommendations codeguru-profiler:ListFindingsReports codeguru-profiler:ListProfileTimes codeguru-profiler:ListProfilingGroups codeguru-profiler:PutPermission codeguru-profiler:RemoveNotificationChannel codeguru-profiler:RemovePermission codeguru-profiler:SubmitFeedback codeguru-profiler:UpdateProfilingGroup

Prefixo do serviço	Ações do
codeguru-reviewer	codeguru-reviewer:AssociateRepository
	codeguru-reviewer:CreateCodeReview
	codeguru-reviewer:DescribeCodeReview
	codeguru-reviewer:DescribeRecommendationFeedback
	codeguru-reviewer:DescribeRepositoryAssociation
	codeguru-reviewer:DisassociateRepository
	codeguru-reviewer:ListCodeReviews
	codeguru-reviewer:ListRecommendationFeedback
	codeguru-reviewer:ListRecommendations
	codeguru-reviewer:ListRepositoryAssociations
	codeguru-reviewer:PutRecommendationFeedback

Prefixo do serviço	Ações do
codepipeline	codepipeline:AcknowledgeJob codepipeline:AcknowledgeThirdPartyJob codepipeline:CreateCustomActionType codepipeline:CreatePipeline codepipeline:DeleteCustomActionType codepipeline:DeletePipeline codepipeline:DeleteWebhook codepipeline:DeregisterWebhookWithThirdParty codepipeline:GetActionType codepipeline:GetJobDetails codepipeline:GetPipeline codepipeline:GetPipelineExecution codepipeline:GetPipelineState codepipeline:GetThirdPartyJobDetails codepipeline:ListActionExecutions codepipeline:ListActionTypes codepipeline:ListPipelineExecutions codepipeline:ListPipelines codepipeline:ListRuleExecutions codepipeline:ListRuleTypes codepipeline:ListWebhooks

Prefixo do serviço	Ações do
	codepipeline:OverrideStageCondition codepipeline:PollForJobs codepipeline:PollForThirdPartyJobs codepipeline:PutActionRevision codepipeline:PutApprovalResult codepipeline:PutJobFailureResult codepipeline:PutJobSuccessResult codepipeline:PutThirdPartyJobFailureResult codepipeline:PutThirdPartyJobSuccessResult codepipeline:PutWebhook codepipeline:RegisterWebhookWithThirdParty codepipeline:RollbackStage codepipeline:StartPipelineExecution codepipeline:StopPipelineExecution codepipeline:UpdateActionType codepipeline:UpdatePipeline

Prefixo do serviço	Ações do
codestar	codestar:AssociateTeamMember
	codestar:CreateProject
	codestar:CreateUserProfile
	codestar:DeleteProject
	codestar:DeleteUserProfile
	codestar:DescribeProject
	codestar:DescribeUserProfile
	codestar:DisassociateTeamMember
	codestar:ListProjects
	codestar:ListResources
	codestar:ListTeamMembers
	codestar:ListUserProfiles
	codestar:UpdateProject
	codestar:UpdateTeamMember
	codestar:UpdateUserProfile

Prefixo do serviço	Ações do
codestar-notifications	codestar-notifications:CreateNotificationRule codestar-notifications:DeleteNotificationRule codestar-notifications:DeleteTarget codestar-notifications:DescribeNotificationRule codestar-notifications:ListEventTypes codestar-notifications:ListNotificationRules codestar-notifications:ListTargets codestar-notifications:Subscribe codestar-notifications:Unsubscribe codestar-notifications:UpdateNotificationRule

Prefixo do serviço	Ações do
cognito-identity	cognito-identity:CreateIdentityPool
	cognito-identity:DeleteIdentities
	cognito-identity:DeleteIdentityPool
	cognito-identity:DescribeIdentity
	cognito-identity:DescribeIdentityPool
	cognito-identity:GetIdentityPoolRoles
	cognito-identity:ListIdentities
	cognito-identity:ListIdentityPools
	cognito-identity:LookupDeveloperIdentity
	cognito-identity:MergeDeveloperIdentities
	cognito-identity:SetIdentityPoolRoles
	cognito-identity:UnlinkDeveloperIdentity
	cognito-identity:UpdateIdentityPool

Prefixo do serviço	Ações do
cognito-idp	cognito-idp:AddCustomAttributes cognito-idp:AddUserPoolClientSecret cognito-idp:AdminAddUserToGroup cognito-idp:AdminConfirmSignUp cognito-idp:AdminCreateUser cognito-idp:AdminDeleteUser cognito-idp:AdminDeleteUserAttributes cognito-idp:AdminDisableProviderForUser cognito-idp:AdminDisableUser cognito-idp:AdminEnableUser cognito-idp:AdminForgetDevice cognito-idp:AdminGetDevice cognito-idp:AdminGetUser cognito-idp:AdminGetUserAuthFactors cognito-idp:AdminInitiateAuth cognito-idp:AdminLinkProviderForUser cognito-idp:AdminListDevices cognito-idp:AdminListGroupsForUser cognito-idp:AdminListUserAuthEvents cognito-idp:AdminRemoveUserFromGroup cognito-idp:AdminResetUserPassword

Prefixo do serviço	Ações do
	cognito-idp:AdminRespondToAuthChallenge cognito-idp:AdminSetUserMFAPreference cognito-idp:AdminSetUserPassword cognito-idp:AdminSetUserSettings cognito-idp:AdminUpdateAuthEventFeedback cognito-idp:AdminUpdateDeviceStatus cognito-idp:AdminUpdateUserAttributes cognito-idp:AdminUserGlobalSignOut cognito-idp:AssociateSoftwareToken cognito-idp:ChangePassword cognito-idp:ConfirmDevice cognito-idp:ConfirmForgotPassword cognito-idp:ConfirmSignUp cognito-idp:CreateGroup cognito-idp:CreateIdentityProvider cognito-idp:CreateManagedLoginBranding cognito-idp:CreateResourceServer cognito-idp:CreateTerms cognito-idp:CreateUserImportJob cognito-idp:CreateUserPool cognito-idp:CreateUserPoolClient

Prefixo do serviço	Ações do
	cognito-idp:CreateUserPoolDomain cognito-idp:CreateUserPoolReplica cognito-idp:DeleteGroup cognito-idp:DeleteIdentityProvider cognito-idp:DeleteManagedLoginBranding cognito-idp:DeleteResourceServer cognito-idp:DeleteTerms cognito-idp:DeleteUser cognito-idp:DeleteUserAttributes cognito-idp:DeleteUserPool cognito-idp:DeleteUserPoolClient cognito-idp:DeleteUserPoolClientSecret cognito-idp:DeleteUserPoolDomain cognito-idp:DeleteUserPoolReplica cognito-idp:DescribeIdentityProvider cognito-idp:DescribeManagedLoginBranding cognito-idp:DescribeManagedLoginBrandingByClient cognito-idp:DescribeResourceServer cognito-idp:DescribeRiskConfiguration cognito-idp:DescribeTerms cognito-idp:DescribeUserImportJob

Prefixo do serviço	Ações do
	cognito-idp:DescribeUserPool cognito-idp:DescribeUserPoolClient cognito-idp:DescribeUserPoolDomain cognito-idp:ForgetDevice cognito-idp:ForgotPassword cognito-idp:GetCSVHeader cognito-idp:GetDevice cognito-idp:GetGroup cognito-idp:GetIdentityProviderByIdentifier cognito-idp:GetLogDeliveryConfiguration cognito-idp:GetProvisionedLimit cognito-idp:GetSigningCertificate cognito-idp:GetUICustomization cognito-idp:GetUser cognito-idp:GetUserAttributeVerificationCode cognito-idp:GetUserPoolMfaConfig cognito-idp:GlobalSignOut cognito-idp:InitiateAuth cognito-idp:ListDevices cognito-idp:ListGroups cognito-idp:ListIdentityProviders

Prefixo do serviço	Ações do
	cognito-idp:ListResourceServers cognito-idp:ListTerms cognito-idp:ListUserImportJobs cognito-idp:ListUserPoolClientSecrets cognito-idp:ListUserPoolClients cognito-idp:ListUserPoolReplicas cognito-idp:ListUserPools cognito-idp:ListUsers cognito-idp:ListUsersInGroup cognito-idp:ResendConfirmationCode cognito-idp:RespondToAuthChallenge cognito-idp:RevokeToken cognito-idp:SetLogDeliveryConfiguration cognito-idp:SetRiskConfiguration cognito-idp:SetUICustomization cognito-idp:SetUserMFAPreference cognito-idp:SetUserPoolMfaConfig cognito-idp:SetUserSettings cognito-idp:SignUp cognito-idp:StartUserImportJob cognito-idp:StopUserImportJob

Prefixo do serviço	Ações do
	cognito-idp:UpdateAuthEventFeedback cognito-idp:UpdateDeviceStatus cognito-idp:UpdateGroup cognito-idp:UpdateIdentityProvider cognito-idp:UpdateProvisionedLimit cognito-idp:UpdateResourceServer cognito-idp:UpdateTerms cognito-idp:UpdateUserAttributes cognito-idp:UpdateUserPool cognito-idp:UpdateUserPoolClient cognito-idp:UpdateUserPoolDomain cognito-idp:UpdateUserPoolReplica cognito-idp:VerifySoftwareToken cognito-idp:VerifyUserAttribute

Prefixo do serviço	Ações do
cognito-sync	cognito-sync:BulkPublish
	cognito-sync:DeleteDataset
	cognito-sync:DescribeDataset
	cognito-sync:DescribeIdentityPoolUsage
	cognito-sync:DescribeIdentityUsage
	cognito-sync:GetBulkPublishDetails
	cognito-sync:GetCognitoEvents
	cognito-sync:GetIdentityPoolConfiguration
	cognito-sync:ListDatasets
	cognito-sync:ListIdentityPoolUsage
	cognito-sync:ListRecords
	cognito-sync:RegisterDevice
	cognito-sync:SetCognitoEvents
	cognito-sync:SetIdentityPoolConfiguration
	cognito-sync:SubscribeToDataset
	cognito-sync:UnsubscribeFromDataset
	cognito-sync:UpdateRecords

Prefixo do serviço	Ações do
comprehendmedical	comprehendmedical:DescribeEntitiesDetectionV2Job comprehendmedical:DescribeICD10CMInferenceJob comprehendmedical:DescribePHIDetectionJob comprehendmedical:DescribeRxNormInferenceJob comprehendmedical:DescribeSNOMEDCTInferenceJob comprehendmedical:DetectEntitiesV2 comprehendmedical:DetectPHI comprehendmedical:InferICD10CM comprehendmedical:InferRxNorm comprehendmedical:InferSNOMEDCT comprehendmedical:ListEntitiesDetectionV2Jobs comprehendmedical:ListICD10CMInferenceJobs comprehendmedical:ListPHIDetectionJobs comprehendmedical:ListRxNormInferenceJobs comprehendmedical:ListSNOMEDCTInferenceJobs comprehendmedical:StartEntitiesDetectionV2Job comprehendmedical:StartICD10CMInferenceJob comprehendmedical:StartPHIDetectionJob comprehendmedical:StartRxNormInferenceJob comprehendmedical:StartSNOMEDCTInferenceJob comprehendmedical:StopEntitiesDetectionV2Job

Prefixo do serviço	Ações do
	comprehendmedical:StopICD10CMInferenceJob comprehendmedical:StopPHIDetectionJob comprehendmedical:StopRxNormInferenceJob comprehendmedical:StopSNOMEDCTInferenceJob

Prefixo do serviço	Ações do
compute-optimizer	compute-optimizer:DeleteRecommendationPreferences
	compute-optimizer:DescribeRecommendationExportJobs
	compute-optimizer:ExportAutoScalingGroupRecommendations
	compute-optimizer:ExportEBSVolumeRecommendations
	compute-optimizer:ExportEC2InstanceRecommendations
	compute-optimizer:ExportECSServiceRecommendations
	compute-optimizer:ExportIdleRecommendations
	compute-optimizer:ExportLambdaFunctionRecommendations
	compute-optimizer:ExportLicenseRecommendations
	compute-optimizer:ExportRDSDatabaseRecommendations
	compute-optimizer:GetEC2RecommendationProjectedMetrics
	compute-optimizer:GetECSServiceRecommendationProjectedMetrics
	compute-optimizer:GetEffectiveRecommendationPreferences
	compute-optimizer:GetEnrollmentStatus
	compute-optimizer:GetEnrollmentStatusesForOrganization
	compute-optimizer:GetRDSDatabaseRecommendationProjectedMetrics
	compute-optimizer:GetRecommendationPreferences
	compute-optimizer:GetRecommendationSummaries
	compute-optimizer:PutRecommendationPreferences

Prefixo do serviço	Ações do
	compute-optimizer:UpdateEnrollmentStatus

Prefixo do serviço	Ações do
config	config:BatchGetResourceConfig config:DeleteAggregationAuthorization config:DeleteConfigRule config:DeleteConfigurationAggregator config:DeleteConfigurationRecorder config:DeleteConformancePack config:DeleteDeliveryChannel config:DeleteEvaluationResults config:DeleteOrganizationConfigRule config:DeleteOrganizationConformancePack config:DeletePendingAggregationRequest config:DeleteRemediationConfiguration config:DeleteRemediationExceptions config:DeleteResourceConfig config:DeleteRetentionConfiguration config:DeleteStoredQuery config:DeliverConfigSnapshot config:DescribeAggregateComplianceByConfigRules config:DescribeAggregateComplianceByConformancePacks config:DescribeAggregationAuthorizations config:DescribeComplianceByConfigRule

Prefixo do serviço	Ações do
	config:DescribeComplianceByResource config:DescribeConfigRuleEvaluationStatus config:DescribeConfigRules config:DescribeConfigurationAggregatorSourcesStatus config:DescribeConfigurationAggregators config:DescribeConfigurationRecorderStatus config:DescribeConfigurationRecorders config:DescribeConformancePackCompliance config:DescribeConformancePackStatus config:DescribeConformancePacks config:DescribeDeliveryChannelStatus config:DescribeDeliveryChannels config:DescribeOrganizationConfigRuleStatuses config:DescribeOrganizationConfigRules config:DescribeOrganizationConformancePackStatuses config:DescribeOrganizationConformancePacks config:DescribePendingAggregationRequests config:DescribeRemediationConfigurations config:DescribeRemediationExceptions config:DescribeRemediationExecutionStatus config:DescribeRetentionConfigurations

Prefixo do serviço	Ações do
	config:GetComplianceDetailsByConfigRule config:GetComplianceDetailsByResource config:GetComplianceSummaryByConfigRule config:GetComplianceSummaryByResourceType config:GetConformancePackComplianceDetails config:GetConformancePackComplianceSummary config:GetCustomRulePolicy config:GetDiscoveredResourceCounts config:GetOrganizationConfigRuleDetailedStatus config:GetOrganizationConformancePackDetailedStatus config:GetOrganizationCustomRulePolicy config:GetResourceConfigHistory config:GetResourceEvaluationSummary config:GetStoredQuery config:ListConfigurationRecorders config:ListConformancePackComplianceScores config:ListConnectors config:ListDiscoveredResources config:ListResourceEvaluations config:ListStoredQueries config:PutConfigRule

Prefixo do serviço	Ações do
	config:PutConfigurationAggregator config:PutConfigurationRecorder config:PutConformancePack config:PutConnector config:PutDeliveryChannel config:PutEvaluations config:PutExternalEvaluation config:PutOrganizationConfigRule config:PutOrganizationConformancePack config:PutRemediationConfigurations config:PutRemediationExceptions config:PutResourceConfig config:PutRetentionConfiguration config:PutStoredQuery config:SelectResourceConfig config:StartConfigRulesEvaluation config:StartConfigurationRecorder config:StartRemediationExecution config:StartResourceEvaluation config:StopConfigurationRecorder

Prefixo do serviço	Ações do
conectar	connect:ActivateEvaluationForm
	connect:AssociateAnalyticsDataSet
	connect:AssociateApprovedOrigin
	connect:AssociateBot
	connect:AssociateContactWithUser
	connect:AssociateDefaultVocabulary
	connect:AssociateEmailAddressAlias
	connect:AssociateFlow
	connect:AssociateInstanceStorageConfig
	connect:AssociateLambdaFunction
	connect:AssociateLexBot
	connect:AssociatePhoneNumberContactFlow
	connect:AssociateQueueEmailAddresses
	connect:AssociateQueueQuickConnects
	connect:AssociateRoutingProfileQueues
	connect:AssociateSecurityKey
	connect:AssociateUserProficiencies
	connect:BatchAssociateAnalyticsDataSet
	connect:BatchCreateDataTableValue
	connect:BatchDeleteDataTableValue
	connect:BatchDescribeDataTableValue

Prefixo do serviço	Ações do
	connect:BatchDisassociateAnalyticsDataSet
	connect:BatchGetFlowAssociation
	connect:BatchPutContact
	connect:BatchUpdateDataTableValue
	connect:ClaimPhoneNumber
	connect:CreateAgentStatus
	connect:CreateAttachedFile
	connect:CreateContact
	connect:CreateContactFlow
	connect:CreateContactFlowModule
	connect:CreateContactFlowModuleAlias
	connect:CreateContactFlowModuleVersion
	connect:CreateContactFlowVersion
	connect:CreateDataTable
	connect:CreateDataTableAttribute
	connect:CreateEmailAddress
	connect:CreateEvaluationForm
	connect:CreateHoursOfOperation
	connect:CreateInstance
	connect:CreateIntegrationAssociation
	connect:CreateParticipant

Prefixo do serviço	Ações do
	connect:CreatePersistentContactAssociation
	connect:CreatePredefinedAttribute
	connect:CreatePrompt
	connect:CreatePushNotificationRegistration
	connect:CreateQueue
	connect:CreateQuickConnect
	connect:CreateRoutingProfile
	connect:CreateRule
	connect:CreateSecurityProfile
	connect:CreateTaskTemplate
	connect:CreateTrafficDistributionGroup
	connect:CreateUseCase
	connect:CreateUser
	connect:CreateUserHierarchyGroup
	connect:CreateView
	connect:CreateViewVersion
	connect:CreateVocabulary
	connect:CreateWorkspace
	connect:DeactivateEvaluationForm
	connect:DeleteContactEvaluation
	connect:DeleteContactFlow

Prefixo do serviço	Ações do
	connect:DeleteContactFlowModule
	connect:DeleteContactFlowModuleAlias
	connect:DeleteContactFlowModuleVersion
	connect:DeleteContactFlowVersion
	connect:DeleteDataTable
	connect:DeleteDataTableAttribute
	connect:DeleteEmailAddress
	connect:DeleteEvaluationForm
	connect:DeleteHoursOfOperation
	connect:DeleteHoursOfOperationOverride
	connect:DeleteInstance
	connect:DeleteIntegrationAssociation
	connect:DeleteNotification
	connect:DeletePredefinedAttribute
	connect:DeletePrompt
	connect:DeletePushNotificationRegistration
	connect:DeleteQueue
	connect:DeleteQuickConnect
	connect:DeleteRoutingProfile
	connect:DeleteRule
	connect:DeleteSecurityProfile

Prefixo do serviço	Ações do
	connect:DeleteTaskTemplate
	connect:DeleteTrafficDistributionGroup
	connect:DeleteUseCase
	connect:DeleteUser
	connect:DeleteUserHierarchyGroup
	connect:DeleteView
	connect:DeleteVocabulary
	connect:DeleteWorkspace
	connect:DeleteWorkspaceMedia
	connect:DescribeAttachedFilesConfiguration
	connect:DescribeAuthenticationProfile
	connect:DescribeContactFlowModuleAlias
	connect:DescribeDataTableAttribute
	connect:DescribeHoursOfOperationOverride
	connect:DescribeInstanceAttribute
	connect:DescribeInstanceStorageConfig
	connect:DescribeNotification
	connect:DescribePhoneNumber
	connect:DescribeRule
	connect:DescribeTrafficDistributionGroup
	connect:DescribeUserHierarchyStructure

Prefixo do serviço	Ações do
	connect:DescribeVocabulary
	connect:DisassociateAnalyticsDataSet
	connect:DisassociateApprovedOrigin
	connect:DisassociateBot
	connect:DisassociateEmailAddressAlias
	connect:DisassociateFlow
	connect:DisassociateInstanceStorageConfig
	connect:DisassociateLambdaFunction
	connect:DisassociateLexBot
	connect:DisassociatePhoneNumberContactFlow
	connect:DisassociateQueueEmailAddresses
	connect:DisassociateQueueQuickConnects
	connect:DisassociateRoutingProfileQueues
	connect:DisassociateSecurityKey
	connect:DisassociateUserProficiencies
	connect:DismissUserContact
	connect:EvaluateDataTableValues
	connect:GetContactAttributes
	connect:GetContactMetrics
	connect:GetCurrentMetricData
	connect:GetCurrentUserData

Prefixo do serviço	Ações do
	connect:GetEffectiveHoursOfOperations
	connect:GetEvaluationFormValidation
	connect:GetFederationToken
	connect:GetFlowAssociation
	connect:GetMetricData
	connect:GetMetricDataV2
	connect:GetPromptFile
	connect:GetTaskTemplate
	connect:GetTrafficDistribution
	connect:ImportPhoneNumber
	connect:ImportWorkspaceMedia
	connect:ListAnalyticsDataAssociations
	connect:ListAnalyticsDataLakeDataSets
	connect:ListApprovedOrigins
	connect:ListAssociatedContacts
	connect:ListAttachedFilesConfigurations
	connect:ListAuthenticationProfiles
	connect:ListBots
	connect:ListContactEvaluations
	connect:ListContactFlowModuleAliases
	connect:ListContactFlowModuleVersions

Prefixo do serviço	Ações do
	connect:ListContactFlowModules
	connect:ListContactFlowVersions
	connect:ListContactFlows
	connect:ListContactReferences
	connect:ListDataTableAttributes
	connect:ListDataTablePrimaryValues
	connect:ListDataTableValues
	connect:ListDataTables
	connect:ListDefaultVocabularies
	connect:ListEvaluationFormVersions
	connect:ListEvaluationForms
	connect:ListFlowAssociations
	connect:ListHoursOfOperations
	connect:ListInstanceAttributes
	connect:ListInstanceStorageConfigs
	connect:ListIntegrationAssociations
	connect:ListLambdaFunctions
	connect:ListLexBots
	connect:ListNotifications
	connect:ListPhoneNumbers
	connect:ListPhoneNumbersV2

Prefixo do serviço	Ações do
	connect:ListPredefinedAttributes
	connect:ListPrompts
	connect:ListQueueEmailAddresses
	connect:ListQueueQuickConnects
	connect:ListQueues
	connect:ListQuickConnects
	connect:ListRealtimeContactAnalysisSegmentsV2
	connect:ListRoutingProfileManualAssignmentQueues
	connect:ListRoutingProfileQueues
	connect:ListRoutingProfiles
	connect:ListRules
	connect:ListSecurityKeys
	connect:ListSecurityProfileApplications
	connect:ListSecurityProfileFlowModules
	connect:ListSecurityProfilePermissions
	connect:ListSecurityProfiles
	connect:ListTaskTemplates
	connect:ListTrafficDistributionGroups
	connect:ListUseCases
	connect:ListUserHierarchyGroups
	connect:ListUserNotifications

Prefixo do serviço	Ações do
	connect:ListUsers
	connect:ListViewVersions
	connect:ListViews
	connect:ListWorkspaceMedia
	connect:ListWorkspacePages
	connect:ListWorkspaces
	connect:MonitorContact
	connect:PauseContact
	connect:PutUserStatus
	connect:ReleasePhoneNumber
	connect:ReplicateInstance
	connect:ResumeContact
	connect:ResumeContactRecording
	connect:SearchAgentStatuses
	connect:SearchAvailablePhoneNumbers
	connect:SearchContactEvaluations
	connect:SearchContactFlowModules
	connect:SearchContactFlows
	connect:SearchContacts
	connect:SearchDataTables
	connect:SearchEmailAddresses

Prefixo do serviço	Ações do
	connect:SearchEvaluationForms
	connect:SearchHoursOfOperations
	connect:SearchPredefinedAttributes
	connect:SearchPrompts
	connect:SearchQueues
	connect:SearchQuickConnects
	connect:SearchRoutingProfiles
	connect:SearchRules
	connect:SearchSecurityProfiles
	connect:SearchUserHierarchyGroups
	connect:SearchViews
	connect:SearchVocabularies
	connect:SearchWorkspaceAssociations
	connect:SearchWorkspaces
	connect:SendChatIntegrationEvent
	connect:SendOutboundEmail
	connect:StartChatContact
	connect:StartContactConversationalAnalyticsJob
	connect:StartContactEvaluation
	connect:StartContactMediaProcessing
	connect:StartContactRecording

Prefixo do serviço	Ações do
	connect:StartContactStreaming
	connect:StartEmailContact
	connect:StartOutboundChatContact
	connect:StartOutboundEmailContact
	connect:StartOutboundVoiceContact
	connect:StartScreenSharing
	connect:StartTaskContact
	connect:StartWebRTCContact
	connect:StopContact
	connect:StopContactMediaProcessing
	connect:StopContactRecording
	connect:StopContactStreaming
	connect:SubmitContactEvaluation
	connect:SuspendContactRecording
	connect:TransferContact
	connect:UpdateAgentStatus
	connect:UpdateAttachedFilesConfiguration
	connect:UpdateAuthenticationProfile
	connect:UpdateContact
	connect:UpdateContactAttributes
	connect:UpdateContactEvaluation

Prefixo do serviço	Ações do
	connect:UpdateContactFlowContent
	connect:UpdateContactFlowMetadata
	connect:UpdateContactFlowModuleAlias
	connect:UpdateContactFlowModuleContent
	connect:UpdateContactFlowModuleMetadata
	connect:UpdateContactFlowName
	connect:UpdateContactRoutingData
	connect:UpdateContactSchedule
	connect:UpdateDataTableAttribute
	connect:UpdateDataTableMetadata
	connect:UpdateDataTablePrimaryValues
	connect:UpdateEmailAddressMetadata
	connect:UpdateEvaluationForm
	connect:UpdateHoursOfOperation
	connect:UpdateHoursOfOperationOverride
	connect:UpdateInstanceAttribute
	connect:UpdateInstanceStorageConfig
	connect:UpdateNotificationContent
	connect:UpdateParticipantAuthentication
	connect:UpdateParticipantRoleConfig
	connect:UpdatePhoneNumber

Prefixo do serviço	Ações do
	connect:UpdatePhoneNumberMetadata
	connect:UpdatePredefinedAttribute
	connect:UpdatePrompt
	connect:UpdateQueueHoursOfOperation
	connect:UpdateQueueMaxContacts
	connect:UpdateQueueName
	connect:UpdateQueueOutboundCallerConfig
	connect:UpdateQueueOutboundEmailConfig
	connect:UpdateQueueStatus
	connect:UpdateQuickConnectConfig
	connect:UpdateQuickConnectName
	connect:UpdateRoutingProfileAgentAvailabilityTimer
	connect:UpdateRoutingProfileConcurrency
	connect:UpdateRoutingProfileDefaultOutboundQueue
	connect:UpdateRoutingProfileName
	connect:UpdateRoutingProfileQueues
	connect:UpdateRule
	connect:UpdateSecurityProfile
	connect:UpdateTaskTemplate
	connect:UpdateTrafficDistribution
	connect:UpdateUserHierarchy

Prefixo do serviço	Ações do
	connect:UpdateUserHierarchyGroupName connect:UpdateUserHierarchyStructure connect:UpdateUserIdentityInfo connect:UpdateUserNotificationStatus connect:UpdateUserPhoneConfig connect:UpdateUserProficiencies connect:UpdateUserRoutingProfile connect:UpdateUserSecurityProfiles connect:UpdateViewContent connect:UpdateViewMetadata connect:UpdateWorkspaceMetadata connect:UpdateWorkspaceTheme connect:UpdateWorkspaceVisibility
cur	cur:DeleteReportDefinition cur:DescribeReportDefinitions cur:ModifyReportDefinition cur:PutReportDefinition

Prefixo do serviço	Ações do
databrew	databrew:BatchDeleteRecipeVersion
	databrew:CreateDataset
	databrew:CreateProfileJob
	databrew:CreateProject
	databrew:CreateRecipe
	databrew:CreateRecipeJob
	databrew:CreateRuleset
	databrew:CreateSchedule
	databrew:DeleteDataset
	databrew:DeleteJob
	databrew:DeleteProject
	databrew:DeleteRecipeVersion
	databrew:DeleteRuleset
	databrew:DeleteSchedule
	databrew:DescribeDataset
	databrew:DescribeJob
	databrew:DescribeJobRun
	databrew:DescribeProject
	databrew:DescribeRecipe
	databrew:DescribeRuleset
	databrew:DescribeSchedule

Prefixo do serviço	Ações do
	databrew:ListDatasets
	databrew:ListJobRuns
	databrew:ListJobs
	databrew:ListProjects
	databrew:ListRecipeVersions
	databrew:ListRecipes
	databrew:ListRulesets
	databrew:ListSchedules
	databrew:PublishRecipe
	databrew:SendProjectSessionAction
	databrew:StartJobRun
	databrew:StartProjectSession
	databrew:StopJobRun
	databrew:UpdateDataset
	databrew:UpdateProfileJob
	databrew:UpdateProject
	databrew:UpdateRecipe
	databrew:UpdateRecipeJob
	databrew:UpdateRuleset
	databrew:UpdateSchedule

Prefixo do serviço	Ações do
dataexchange	dataexchange:AcceptDataGrant dataexchange:CancelJob dataexchange:CreateDataGrant dataexchange:CreateDataSet dataexchange:CreateEventAction dataexchange:CreateJob dataexchange:CreateRevision dataexchange:DeleteAsset dataexchange:DeleteDataGrant dataexchange:DeleteEventAction dataexchange:DeleteRevision dataexchange:GetDataGrant dataexchange:GetEventAction dataexchange:GetJob dataexchange:GetReceivedDataGrant dataexchange:ListDataGrants dataexchange:ListDataSetRevisions dataexchange:ListDataSets dataexchange:ListEventActions dataexchange:ListJobs dataexchange:ListReceivedDataGrants

Prefixo do serviço	Ações do
	dataexchange:ListRevisionAssets dataexchange:RevokeRevision dataexchange:SendDataSetNotification dataexchange:StartJob dataexchange:UpdateAsset dataexchange:UpdateDataSet dataexchange:UpdateEventAction dataexchange:UpdateRevision

Prefixo do serviço	Ações do
datapipeline	datapipeline:ActivatePipeline
	datapipeline:CreatePipeline
	datapipeline:DeactivatePipeline
	datapipeline:DeletePipeline
	datapipeline:DescribeObjects
	datapipeline:DescribePipelines
	datapipeline:EvaluateExpression
	datapipeline:GetPipelineDefinition
	datapipeline:ListPipelines
	datapipeline:PollForTask
	datapipeline:PutPipelineDefinition
	datapipeline:QueryObjects
	datapipeline:ReportTaskProgress
	datapipeline:ReportTaskRunnerHeartbeat
	datapipeline:SetStatus
	datapipeline:SetTaskStatus
	datapipeline:ValidatePipelineDefinition

Prefixo do serviço	Ações do
dax	dax:CreateCluster dax:DecreaseReplicationFactor dax:DeleteCluster dax:DeleteParameterGroup dax:DeleteSubnetGroup dax:DescribeClusters dax:DescribeDefaultParameters dax:DescribeEvents dax:DescribeParameterGroups dax:DescribeParameters dax:DescribeSubnetGroups dax:IncreaseReplicationFactor dax:RebootNode dax:UpdateCluster dax:UpdateParameterGroup dax:UpdateSubnetGroup

Prefixo do serviço	Ações do
devicefarm	devicefarm:CreateDevicePool
	devicefarm:CreateInstanceProfile
	devicefarm:CreateNetworkProfile
	devicefarm:CreateProject
	devicefarm:CreateRemoteAccessSession
	devicefarm:CreateTestGridProject
	devicefarm:CreateTestGridUrl
	devicefarm:CreateUpload
	devicefarm:CreateVPCEConfiguration
	devicefarm:DeleteDevicePool
	devicefarm:DeleteInstanceProfile
	devicefarm:DeleteNetworkProfile
	devicefarm:DeleteProject
	devicefarm:DeleteRemoteAccessSession
	devicefarm:DeleteRun
	devicefarm:DeleteTestGridProject
	devicefarm:DeleteUpload
	devicefarm:DeleteVPCEConfiguration
	devicefarm:GetAccountSettings
	devicefarm:GetDevice
	devicefarm:GetDeviceInstance

Prefixo do serviço	Ações do
	devicefarm:GetDevicePool
	devicefarm:GetDevicePoolCompatibility
	devicefarm:GetInstanceProfile
	devicefarm:GetJob
	devicefarm:GetNetworkProfile
	devicefarm:GetOfferingStatus
	devicefarm:GetProject
	devicefarm:GetRemoteAccessSession
	devicefarm:GetRun
	devicefarm:GetSuite
	devicefarm:GetTest
	devicefarm:GetTestGridProject
	devicefarm:GetTestGridSession
	devicefarm:GetUpload
	devicefarm:GetVPCEConfiguration
	devicefarm:ListArtifacts
	devicefarm:ListDeviceInstances
	devicefarm:ListDevicePools
	devicefarm:ListDevices
	devicefarm:ListInstanceProfiles
	devicefarm:ListJobs

Prefixo do serviço	Ações do
	devicefarm:ListNetworkProfiles
	devicefarm:ListOfferingPromotions
	devicefarm:ListOfferingTransactions
	devicefarm:ListOfferings
	devicefarm:ListProjects
	devicefarm:ListRemoteAccessSessions
	devicefarm:ListRuns
	devicefarm:ListSamples
	devicefarm:ListSuites
	devicefarm:ListTestGridProjects
	devicefarm:ListTestGridSessionActions
	devicefarm:ListTestGridSessionArtifacts
	devicefarm:ListTestGridSessions
	devicefarm:ListTests
	devicefarm:ListUniqueProblems
	devicefarm:ListUploads
	devicefarm:ListVPCEConfigurations
	devicefarm:PurchaseOffering
	devicefarm:RenewOffering
	devicefarm:ScheduleRun
	devicefarm:StopJob

Prefixo do serviço	Ações do
	devicefarm:StopRemoteAccessSession
	devicefarm:StopRun
	devicefarm:UpdateDeviceInstance
	devicefarm:UpdateDevicePool
	devicefarm:UpdateInstanceProfile
	devicefarm:UpdateNetworkProfile
	devicefarm:UpdateProject
	devicefarm:UpdateTestGridProject
	devicefarm:UpdateUpload
	devicefarm:UpdateVPCEConfiguration

Prefixo do serviço	Ações do
devops-guru	devops-guru:AddNotificationChannel devops-guru:DeleteInsight devops-guru:DescribeAccountHealth devops-guru:DescribeAccountOverview devops-guru:DescribeAnomaly devops-guru:DescribeEventSourcesConfig devops-guru:DescribeFeedback devops-guru:DescribeInsight devops-guru:DescribeOrganizationHealth devops-guru:DescribeOrganizationOverview devops-guru:DescribeOrganizationResourceCollectionHealth devops-guru:DescribeResourceCollectionHealth devops-guru:DescribeServiceIntegration devops-guru:GetCostEstimation devops-guru:GetResourceCollection devops-guru:ListAnomaliesForInsight devops-guru:ListAnomalousLogGroups devops-guru:ListEvents devops-guru:ListInsights devops-guru:ListMonitoredResources devops-guru:ListNotificationChannels

Prefixo do serviço	Ações do
	devops-guru:ListOrganizationInsights devops-guru:ListRecommendations devops-guru:PutFeedback devops-guru:RemoveNotificationChannel devops-guru:SearchInsights devops-guru:SearchOrganizationInsights devops-guru:StartCostEstimation devops-guru:UpdateEventSourcesConfig devops-guru:UpdateResourceCollection devops-guru:UpdateServiceIntegration

Prefixo do serviço	Ações do
directconnect	directconnect:AcceptDirectConnectGatewayAssociationProposal
	directconnect:AllocateConnectionOnInterconnect
	directconnect:AllocateHostedConnection
	directconnect:AllocatePrivateVirtualInterface
	directconnect:AllocatePublicVirtualInterface
	directconnect:AllocateTransitVirtualInterface
	directconnect:AssociateConnectionWithLag
	directconnect:AssociateHostedConnection
	directconnect:AssociateMacSecKey
	directconnect:AssociateVirtualInterface
	directconnect:ConfirmConnection
	directconnect:ConfirmCustomerAgreement
	directconnect:ConfirmPrivateVirtualInterface
	directconnect:ConfirmPublicVirtualInterface
	directconnect:ConfirmTransitVirtualInterface
	directconnect:CreateBGPPeer
	directconnect:CreateConnection
	directconnect:CreateDirectConnectGateway
	directconnect:CreateDirectConnectGatewayAssociation
	directconnect:CreateDirectConnectGatewayAssociationProposal
	directconnect:CreateInterconnect

Prefixo do serviço	Ações do
	directconnect:CreateLag
	directconnect:CreatePrivateVirtualInterface
	directconnect:CreatePublicVirtualInterface
	directconnect:CreateTransitVirtualInterface
	directconnect:DeleteBGPPeer
	directconnect:DeleteConnection
	directconnect:DeleteDirectConnectGateway
	directconnect:DeleteDirectConnectGatewayAssociation
	directconnect:DeleteDirectConnectGatewayAssociationProposal
	directconnect:DeleteInterconnect
	directconnect:DeleteLag
	directconnect:DeleteVirtualInterface
	directconnect:DescribeConnectionLoa
	directconnect:DescribeConnections
	directconnect:DescribeConnectionsOnInterconnect
	directconnect:DescribeCustomerMetadata
	directconnect:DescribeDirectConnectGatewayAssociationProposals
	directconnect:DescribeDirectConnectGatewayAssociations
	directconnect:DescribeDirectConnectGatewayAttachments
	directconnect:DescribeDirectConnectGateways
	directconnect:DescribeHostedConnections

Prefixo do serviço	Ações do
	directconnect:DescribeInterconnectLoa
	directconnect:DescribeInterconnects
	directconnect:DescribeLags
	directconnect:DescribeLoa
	directconnect:DescribeLocations
	directconnect:DescribeRouterConfiguration
	directconnect:DescribeVirtualGateways
	directconnect:DescribeVirtualInterfaces
	directconnect:DisassociateConnectionFromLag
	directconnect:DisassociateMacSecKey
	directconnect:ListVirtualInterfaceTestHistory
	directconnect:StartBgpFailoverTest
	directconnect:StopBgpFailoverTest
	directconnect:UpdateConnection
	directconnect:UpdateDirectConnectGateway
	directconnect:UpdateDirectConnectGatewayAssociation
	directconnect:UpdateLag
	directconnect:UpdateVirtualInterfaceAttributes

Prefixo do serviço	Ações do
dlm	dlm:CreateLifecyclePolicy dlm:DeleteLifecyclePolicy dlm:GetLifecyclePolicies dlm:GetLifecyclePolicy dlm:UpdateLifecyclePolicy

Prefixo do serviço	Ações do
dms	dms:ApplyPendingMaintenanceAction
	dms:AssociateExtensionPack
	dms:BatchStartRecommendations
	dms:CancelMetadataModelCreation
	dms:CancelReplicationTaskAssessmentRun
	dms:CreateDataProvider
	dms:CreateEndpoint
	dms:CreateEventSubscription
	dms:CreateInstanceProfile
	dms:CreateMigrationProject
	dms:CreateReplicationConfig
	dms:CreateReplicationInstance
	dms:CreateReplicationSubnetGroup
	dms:CreateReplicationTask
	dms:DeleteCertificate
	dms:DeleteConnection
	dms:DeleteDataMigration
	dms:DeleteDataProvider
	dms:DeleteEndpoint
	dms:DeleteEventSubscription
	dms:DeleteFleetAdvisorCollector

Prefixo do serviço	Ações do
	dms:DeleteFleetAdvisorDatabases
	dms:DeleteInstanceProfile
	dms:DeleteMigrationProject
	dms:DeleteReplicationConfig
	dms:DeleteReplicationInstance
	dms:DeleteReplicationSubnetGroup
	dms:DeleteReplicationTask
	dms:DeleteReplicationTaskAssessmentRun
	dms:DescribeAccountAttributes
	dms:DescribeApplicableIndividualAssessments
	dms:DescribeCertificates
	dms:DescribeConnections
	dms:DescribeDataMigrations
	dms:DescribeEndpointSettings
	dms:DescribeEndpointTypes
	dms:DescribeEndpoints
	dms:DescribeEngineVersions
	dms:DescribeEventCategories
	dms:DescribeEventSubscriptions
	dms:DescribeEvents
	dms:DescribeFleetAdvisorCollectors

Prefixo do serviço	Ações do
	dms:DescribeFleetAdvisorDatabases
	dms:DescribeFleetAdvisorLsaAnalysis
	dms:DescribeFleetAdvisorSchemaObjectSummary
	dms:DescribeFleetAdvisorSchemas
	dms:DescribeMetadataModel
	dms:DescribeMetadataModelChildren
	dms:DescribeMetadataModelCreations
	dms:DescribeMetadataModelImports
	dms:DescribeOrderableReplicationInstances
	dms:DescribePendingMaintenanceActions
	dms:DescribeRecommendationLimitations
	dms:DescribeRecommendations
	dms:DescribeRefreshSchemasStatus
	dms:DescribeReplicationConfigs
	dms:DescribeReplicationInstanceTaskLogs
	dms:DescribeReplicationInstances
	dms:DescribeReplicationSubnetGroups
	dms:DescribeReplicationTableStatistics
	dms:DescribeReplicationTaskAssessmentResults
	dms:DescribeReplicationTaskAssessmentRuns
	dms:DescribeReplicationTaskIndividualAssessments

Prefixo do serviço	Ações do
	dms:DescribeReplicationTasks
	dms:DescribeReplications
	dms:DescribeSchemas
	dms:DescribeTableStatistics
	dms:ExportMetadataModelAssessment
	dms:ImportCertificate
	dms:ListDataProviders
	dms:ListExtensionPacks
	dms:ListInstanceProfiles
	dms:ListMetadataModelAssessments
	dms:ListMetadataModelConversions
	dms:ListMetadataModelExports
	dms:ListMigrationProjects
	dms:ModifyDataMigration
	dms:ModifyEndpoint
	dms:ModifyEventSubscription
	dms:ModifyReplicationConfig
	dms:ModifyReplicationInstance
	dms:ModifyReplicationSubnetGroup
	dms:ModifyReplicationTask
	dms:MoveReplicationTask

Prefixo do serviço	Ações do
	dms:RebootReplicationInstance
	dms:RefreshSchemas
	dms:ReloadReplicationTables
	dms:ReloadTables
	dms:RunFleetAdvisorLsaAnalysis
	dms:StartMetadataModelAssessment
	dms:StartMetadataModelConversion
	dms:StartMetadataModelCreation
	dms:StartMetadataModelExportAsScripts
	dms:StartMetadataModelExportToTarget
	dms:StartRecommendations
	dms:StartReplication
	dms:StartReplicationTask
	dms:StartReplicationTaskAssessment
	dms:StopDataMigration
	dms:StopReplicationTask
	dms:TestConnection
	dms:UpdateConversionConfiguration
	dms:UpdateDataProvider
	dms:UpdateInstanceProfile
	dms:UpdateMigrationProject

Prefixo do serviço	Ações do
	dms:UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic:ApplyPendingMaintenanceAction docdb-elastic:CopyClusterSnapshot docdb-elastic:DeleteCluster docdb-elastic:DeleteClusterSnapshot docdb-elastic:GetCluster docdb-elastic:GetClusterSnapshot docdb-elastic:GetPendingMaintenanceAction docdb-elastic:ListClusterSnapshots docdb-elastic:ListClusters docdb-elastic:ListPendingMaintenanceActions docdb-elastic:RestoreClusterFromSnapshot docdb-elastic:StartCluster docdb-elastic:StopCluster docdb-elastic:UpdateCluster

Prefixo do serviço	Ações do
dynamodb	dynamodb:CreateBackup
	dynamodb:CreateGlobalTable
	dynamodb:CreateTable
	dynamodb:DeleteBackup
	dynamodb:DeleteTable
	dynamodb:DescribeBackup
	dynamodb:DescribeContinuousBackups
	dynamodb:DescribeContributorInsights
	dynamodb:DescribeEndpoints
	dynamodb:DescribeExport
	dynamodb:DescribeGlobalTable
	dynamodb:DescribeGlobalTableSettings
	dynamodb:DescribeImport
	dynamodb:DescribeKinesisStreamingDestination
	dynamodb:DescribeLimits
	dynamodb:DescribeStream
	dynamodb:DescribeTable
	dynamodb:DescribeTableReplicaAutoScaling
	dynamodb:DescribeTimeToLive
	dynamodb:DisableKinesisStreamingDestination
	dynamodb:EnableKinesisStreamingDestination

Prefixo do serviço	Ações do
	dynamodb:ExportTableToPointInTime
	dynamodb:GetResourcePolicy
	dynamodb:ImportTable
	dynamodb:ListBackups
	dynamodb:ListContributorInsights
	dynamodb:ListExports
	dynamodb:ListGlobalTables
	dynamodb:ListImports
	dynamodb:ListStreams
	dynamodb:ListTables
	dynamodb:RestoreTableFromBackup
	dynamodb:RestoreTableToPointInTime
	dynamodb:UpdateContinuousBackups
	dynamodb:UpdateContributorInsights
	dynamodb:UpdateGlobalTable
	dynamodb:UpdateGlobalTableSettings
	dynamodb:UpdateKinesisStreamingDestination
	dynamodb:UpdateTable
	dynamodb:UpdateTableReplicaAutoScaling
	dynamodb:UpdateTimeToLive

Prefixo do serviço	Ações do
ebs	ebs:CompleteSnapshot ebs:StartSnapshot

Prefixo do serviço	Ações do
ec2	ec2:AcceptAddressTransfer
	ec2:AcceptCapacityReservationBillingOwnership
	ec2:AcceptReservedInstancesExchangeQuote
	ec2:AcceptTransitGatewayClientVpnAttachment
	ec2:AcceptTransitGatewayMulticastDomainAssociations
	ec2:AcceptTransitGatewayPeeringAttachment
	ec2:AcceptTransitGatewayVpcAttachment
	ec2:AcceptVpcEndpointConnections
	ec2:AcceptVpcPeeringConnection
	ec2:AdvertiseByoipCidr
	ec2:AllocateAddress
	ec2:AllocateHosts
	ec2:AllocateIpamPoolCidr
	ec2:ApplySecurityGroupsToClientVpnTargetNetwork
	ec2:AssignIpv6Addresses
	ec2:AssignPrivateIpAddresses
	ec2:AssignPrivateNatGatewayAddress
	ec2:AssociateAddress
	ec2:AssociateCapacityReservationBillingOwner
	ec2:AssociateClientVpnTargetNetwork
	ec2:AssociateDhcpOptions

Prefixo do serviço	Ações do
	ec2:AssociateEnclaveCertificateIamRole
	ec2:AssociateIamInstanceProfile
	ec2:AssociateInstanceEventWindow
	ec2:AssociateIpamByoasn
	ec2:AssociateIpamResourceDiscovery
	ec2:AssociateNatGatewayAddress
	ec2:AssociateRouteServer
	ec2:AssociateRouteTable
	ec2:AssociateSecurityGroupVpc
	ec2:AssociateSubnetCidrBlock
	ec2:AssociateTransitGatewayMulticastDomain
	ec2:AssociateTransitGatewayPolicyTable
	ec2:AssociateTransitGatewayRouteTable
	ec2:AssociateTrunkInterface
	ec2:AssociateVpcCidrBlock
	ec2:AttachClassicLinkVpc
	ec2:AttachImageWatermark
	ec2:AttachInternetGateway
	ec2:AttachNetworkInterface
	ec2:AttachVerifiedAccessTrustProvider
	ec2:AttachVolume

Prefixo do serviço	Ações do
	ec2:AttachVpnGateway
	ec2:AuthorizeClientVpnIngress
	ec2:AuthorizeSecurityGroupEgress
	ec2:AuthorizeSecurityGroupIngress
	ec2:BundleInstance
	ec2:CancelBundleTask
	ec2:CancelCapacityReservation
	ec2:CancelCapacityReservationFleets
	ec2:CancelConversionTask
	ec2:CancelDeclarativePoliciesReport
	ec2:CancelExportTask
	ec2:CancelImageLaunchPermission
	ec2:CancelImportTask
	ec2:CancelReservedInstancesListing
	ec2:CancelSpotFleetRequests
	ec2:CancelSpotInstanceRequests
	ec2:ConfirmProductInstance
	ec2:CopyFpgaImage
	ec2:CopyImage
	ec2:CopySnapshot
	ec2:CopyVolumes

Prefixo do serviço	Ações do
	ec2:CreateCapacityManagerDataExport
	ec2:CreateCapacityReservation
	ec2:CreateCapacityReservationBySplitting
	ec2:CreateCapacityReservationCancellationQuote
	ec2:CreateCapacityReservationFleet
	ec2:CreateCarrierGateway
	ec2:CreateClientVpnEndpoint
	ec2:CreateClientVpnRoute
	ec2:CreateCoipCidr
	ec2:CreateCoipPool
	ec2:CreateCustomerGateway
	ec2:CreateDefaultSubnet
	ec2:CreateDefaultVpc
	ec2:CreateDelegateMacVolumeOwnershipTask
	ec2:CreateDhcpOptions
	ec2:CreateEgressOnlyInternetGateway
	ec2:CreateFleet
	ec2:CreateFlowLogs
	ec2:CreateFpgaImage
	ec2:CreateImage
	ec2:CreateImageUsageReport

Prefixo do serviço	Ações do
	ec2:CreateInstanceConnectEndpoint
	ec2:CreateInstanceEventWindow
	ec2:CreateInstanceExportTask
	ec2:CreateInternetGateway
	ec2:CreateInterruptibleCapacityReservationAllocation
	ec2:CreateIpam
	ec2:CreateIpamExternalResourceVerificationToken
	ec2:CreateIpamPolicy
	ec2:CreateIpamPool
	ec2:CreateIpamPrefixListResolver
	ec2:CreateIpamPrefixListResolverTarget
	ec2:CreateIpamResourceDiscovery
	ec2:CreateIpamScope
	ec2:CreateKeyPair
	ec2:CreateLaunchTemplateVersion
	ec2:CreateLocalGatewayRoute
	ec2:CreateLocalGatewayRouteTable
	ec2:CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:CreateLocalGatewayRouteTableVpcAssociation
	ec2:CreateLocalGatewayVirtualInterface

Prefixo do serviço	Ações do
	ec2:CreateLocalGatewayVirtualInterfaceGroup
	ec2:CreateMacSystemIntegrityProtectionModificationTask
	ec2:CreateManagedPrefixList
	ec2:CreateNatGateway
	ec2:CreateNetworkAcl
	ec2:CreateNetworkAclEntry
	ec2:CreateNetworkInsightsAccessScope
	ec2:CreateNetworkInsightsPath
	ec2:CreateNetworkInterface
	ec2:CreateNetworkInterfacePermission
	ec2:CreatePlacementGroup
	ec2:CreatePublicIpv4Pool
	ec2:CreateReplaceRootVolumeTask
	ec2:CreateReservedInstancesListing
	ec2:CreateRestoreImageTask
	ec2:CreateRoute
	ec2:CreateRouteServer
	ec2:CreateRouteServerEndpoint
	ec2:CreateRouteServerPeer
	ec2:CreateRouteTable
	ec2:CreateSecondaryNetwork

Prefixo do serviço	Ações do
	ec2:CreateSecondarySubnet
	ec2:CreateSecurityGroup
	ec2:CreateSnapshots
	ec2:CreateSpotDatafeedSubscription
	ec2:CreateStoreImageTask
	ec2:CreateSubnet
	ec2:CreateSubnetCidrReservation
	ec2:CreateTrafficMirrorFilter
	ec2:CreateTrafficMirrorFilterRule
	ec2:CreateTrafficMirrorSession
	ec2:CreateTrafficMirrorTarget
	ec2:CreateTransitGateway
	ec2:CreateTransitGatewayConnect
	ec2:CreateTransitGatewayConnectPeer
	ec2:CreateTransitGatewayMeteringPolicy
	ec2:CreateTransitGatewayMeteringPolicyEntry
	ec2:CreateTransitGatewayMulticastDomain
	ec2:CreateTransitGatewayPeeringAttachment
	ec2:CreateTransitGatewayPolicyTable
	ec2:CreateTransitGatewayPrefixListReference
	ec2:CreateTransitGatewayRoute

Prefixo do serviço	Ações do
	ec2:CreateTransitGatewayRouteTable
	ec2:CreateTransitGatewayRouteTableAnnouncement
	ec2:CreateTransitGatewayVpcAttachment
	ec2:CreateVerifiedAccessEndpoint
	ec2:CreateVerifiedAccessGroup
	ec2:CreateVerifiedAccessInstance
	ec2:CreateVerifiedAccessTrustProvider
	ec2:CreateVolume
	ec2:CreateVpc
	ec2:CreateVpcBlockPublicAccessExclusion
	ec2:CreateVpcEncryptionControl
	ec2:CreateVpcEndpoint
	ec2:CreateVpcEndpointConnectionNotification
	ec2:CreateVpcEndpointServiceConfiguration
	ec2:CreateVpcPeeringConnection
	ec2:CreateVpnConcentrator
	ec2:CreateVpnConnection
	ec2:CreateVpnConnectionRoute
	ec2:CreateVpnGateway
	ec2:DeleteCapacityManagerDataExport
	ec2:DeleteCarrierGateway

Prefixo do serviço	Ações do
	ec2:DeleteClientVpnEndpoint
	ec2:DeleteClientVpnRoute
	ec2:DeleteCoipCidr
	ec2:DeleteCoipPool
	ec2:DeleteCustomerGateway
	ec2:DeleteDhcpOptions
	ec2:DeleteEgressOnlyInternetGateway
	ec2:DeleteFleets
	ec2:DeleteFlowLogs
	ec2:DeleteFpgaImage
	ec2:DeleteImageUsageReport
	ec2:DeleteInstanceConnectEndpoint
	ec2:DeleteInstanceEventWindow
	ec2:DeleteInternetGateway
	ec2:DeleteIpam
	ec2:DeleteIpamExternalResourceVerificationToken
	ec2:DeleteIpamPolicy
	ec2:DeleteIpamPool
	ec2:DeleteIpamPrefixListResolver
	ec2:DeleteIpamPrefixListResolverTarget
	ec2:DeleteIpamResourceDiscovery

Prefixo do serviço	Ações do
	ec2:DeleteIpamScope
	ec2:DeleteKeyPair
	ec2:DeleteLaunchTemplate
	ec2:DeleteLaunchTemplateVersions
	ec2:DeleteLocalGatewayRoute
	ec2:DeleteLocalGatewayRouteTable
	ec2:DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:DeleteLocalGatewayRouteTableVpcAssociation
	ec2:DeleteLocalGatewayVirtualInterface
	ec2:DeleteLocalGatewayVirtualInterfaceGroup
	ec2:DeleteManagedPrefixList
	ec2:DeleteNatGateway
	ec2:DeleteNetworkAcl
	ec2:DeleteNetworkAclEntry
	ec2:DeleteNetworkInsightsAccessScope
	ec2:DeleteNetworkInsightsAccessScopeAnalysis
	ec2:DeleteNetworkInsightsAnalysis
	ec2:DeleteNetworkInsightsPath
	ec2:DeleteNetworkInterface
	ec2:DeleteNetworkInterfacePermission

Prefixo do serviço	Ações do
	ec2:DeletePlacementGroup
	ec2:DeletePublicIpv4Pool
	ec2:DeleteQueuedReservedInstances
	ec2:DeleteRoute
	ec2:DeleteRouteServer
	ec2:DeleteRouteServerEndpoint
	ec2:DeleteRouteServerPeer
	ec2:DeleteRouteTable
	ec2:DeleteSecondaryNetwork
	ec2:DeleteSecondarySubnet
	ec2:DeleteSecurityGroup
	ec2:DeleteSpotDatafeedSubscription
	ec2:DeleteSubnet
	ec2:DeleteSubnetCidrReservation
	ec2:DeleteTrafficMirrorFilter
	ec2:DeleteTrafficMirrorFilterRule
	ec2:DeleteTrafficMirrorSession
	ec2:DeleteTrafficMirrorTarget
	ec2:DeleteTransitGateway
	ec2:DeleteTransitGatewayClientVpnAttachment
	ec2:DeleteTransitGatewayConnect

Prefixo do serviço	Ações do
	ec2:DeleteTransitGatewayConnectPeer
	ec2:DeleteTransitGatewayMeteringPolicy
	ec2:DeleteTransitGatewayMeteringPolicyEntry
	ec2:DeleteTransitGatewayMulticastDomain
	ec2:DeleteTransitGatewayPeeringAttachment
	ec2:DeleteTransitGatewayPolicyTable
	ec2:DeleteTransitGatewayPrefixListReference
	ec2:DeleteTransitGatewayRoute
	ec2:DeleteTransitGatewayRouteTable
	ec2:DeleteTransitGatewayRouteTableAnnouncement
	ec2:DeleteTransitGatewayVpcAttachment
	ec2:DeleteVerifiedAccessEndpoint
	ec2:DeleteVerifiedAccessGroup
	ec2:DeleteVerifiedAccessInstance
	ec2:DeleteVerifiedAccessTrustProvider
	ec2:DeleteVolume
	ec2:DeleteVpc
	ec2:DeleteVpcBlockPublicAccessExclusion
	ec2:DeleteVpcEncryptionControl
	ec2:DeleteVpcEndpointConnectionNotifications
	ec2:DeleteVpcEndpointServiceConfigurations

Prefixo do serviço	Ações do
	ec2:DeleteVpcEndpoints
	ec2:DeleteVpcPeeringConnection
	ec2:DeleteVpnConcentrator
	ec2:DeleteVpnConnection
	ec2:DeleteVpnConnectionRoute
	ec2:DeleteVpnGateway
	ec2:DeprovisionByoipCidr
	ec2:DeprovisionIpamByoasn
	ec2:DeprovisionIpamPoolCidr
	ec2:DeprovisionPublicIpv4PoolCidr
	ec2:DeregisterImage
	ec2:DeregisterInstanceEventNotificationAttributes
	ec2:DeregisterTransitGatewayMulticastGroupMembers
	ec2:DeregisterTransitGatewayMulticastGroupSources
	ec2:DescribeAccountAttributes
	ec2:DescribeAddressTransfers
	ec2:DescribeAddresses
	ec2:DescribeAddressesAttribute
	ec2:DescribeAggregateIdFormat
	ec2:DescribeAvailabilityZones
	ec2:DescribeAwsNetworkPerformanceMetricSubscriptions

Prefixo do serviço	Ações do
	ec2:DescribeBundleTasks
	ec2:DescribeByoipCidrs
	ec2:DescribeCapacityBlockExtensionHistory
	ec2:DescribeCapacityBlockExtensionOfferings
	ec2:DescribeCapacityBlockStatus
	ec2:DescribeCapacityBlocks
	ec2:DescribeCapacityManagerDataExports
	ec2:DescribeCapacityReservationBillingRequests
	ec2:DescribeCapacityReservationCancellationQuotes
	ec2:DescribeCapacityReservationFleets
	ec2:DescribeCapacityReservationTopology
	ec2:DescribeCapacityReservations
	ec2:DescribeCarrierGateways
	ec2:DescribeClassicLinkInstances
	ec2:DescribeClientVpnAuthorizationRules
	ec2:DescribeClientVpnConnections
	ec2:DescribeClientVpnEndpoints
	ec2:DescribeClientVpnRoutes
	ec2:DescribeClientVpnTargetNetworks
	ec2:DescribeCoipPools
	ec2:DescribeConversionTasks

Prefixo do serviço	Ações do
	ec2:DescribeCustomerGateways
	ec2:DescribeDeclarativePoliciesReports
	ec2:DescribeDhcpOptions
	ec2:DescribeEgressOnlyInternetGateways
	ec2:DescribeElasticGpus
	ec2:DescribeExportImageTasks
	ec2:DescribeExportTasks
	ec2:DescribeFastLaunchImages
	ec2:DescribeFastSnapshotRestores
	ec2:DescribeFleetHistory
	ec2:DescribeFleetInstances
	ec2:DescribeFleets
	ec2:DescribeFlowLogs
	ec2:DescribeFpgaImageAttribute
	ec2:DescribeFpgaImages
	ec2:DescribeHostReservationOfferings
	ec2:DescribeHostReservations
	ec2:DescribeHosts
	ec2:DescribeIamInstanceProfileAssociations
	ec2:DescribeIdFormat
	ec2:DescribeIdentityIdFormat

Prefixo do serviço	Ações do
	ec2:DescribeImageAttribute
	ec2:DescribeImageReferences
	ec2:DescribeImageUsageReportEntries
	ec2:DescribeImageUsageReports
	ec2:DescribeImportImageTasks
	ec2:DescribeImportSnapshotTasks
	ec2:DescribeInstanceConnectEndpoints
	ec2:DescribeInstanceCreditSpecifications
	ec2:DescribeInstanceEventNotificationAttributes
	ec2:DescribeInstanceEventWindows
	ec2:DescribeInstanceImageMetadata
	ec2:DescribeInstanceSqlHaHistoryStates
	ec2:DescribeInstanceSqlHaStates
	ec2:DescribeInstanceTopology
	ec2:DescribeInstanceTypes
	ec2:DescribeInternetGateways
	ec2:DescribeIpamByoasn
	ec2:DescribeIpamExternalResourceVerificationTokens
	ec2:DescribeIpamPolicies
	ec2:DescribeIpamPoolAllocations
	ec2:DescribeIpamPools

Prefixo do serviço	Ações do
	ec2:DescribeIpamPrefixListResolverTargets
	ec2:DescribeIpamPrefixListResolvers
	ec2:DescribeIpamResourceDiscoveries
	ec2:DescribeIpamResourceDiscoveryAssociations
	ec2:DescribeIpamScopes
	ec2:DescribeIpams
	ec2:DescribeIpv6Pools
	ec2:DescribeKeyPairs
	ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	ec2:DescribeLocalGatewayRouteTableVpcAssociations
	ec2:DescribeLocalGatewayRouteTables
	ec2:DescribeLocalGatewayVirtualInterfaceGroups
	ec2:DescribeLocalGatewayVirtualInterfaces
	ec2:DescribeLocalGateways
	ec2:DescribeLockedSnapshots
	ec2:DescribeMacHosts
	ec2:DescribeMacModificationTasks
	ec2:DescribeManagedPrefixLists
	ec2:DescribeMovingAddresses
	ec2:DescribeNatGateways

Prefixo do serviço	Ações do
	ec2:DescribeNetworkAcls
	ec2:DescribeNetworkInsightsAccessScopeAnalyses
	ec2:DescribeNetworkInsightsAccessScopes
	ec2:DescribeNetworkInsightsAnalyses
	ec2:DescribeNetworkInsightsPaths
	ec2:DescribeNetworkInterfaceAttribute
	ec2:DescribeNetworkInterfacePermissions
	ec2:DescribeNetworkInterfaces
	ec2:DescribeOutpostLags
	ec2:DescribePlacementGroups
	ec2:DescribePrefixLists
	ec2:DescribePrincipalIdFormat
	ec2:DescribePublicIpv4Pools
	ec2:DescribeRegions
	ec2:DescribeReplaceRootVolumeTasks
	ec2:DescribeReservedInstances
	ec2:DescribeReservedInstancesListings
	ec2:DescribeReservedInstancesModifications
	ec2:DescribeReservedInstancesOfferings
	ec2:DescribeRouteServerEndpoints
	ec2:DescribeRouteServerPeers

Prefixo do serviço	Ações do
	ec2:DescribeRouteServers
	ec2:DescribeRouteTables
	ec2:DescribeScheduledInstanceAvailability
	ec2:DescribeScheduledInstances
	ec2:DescribeSecondaryInterfaces
	ec2:DescribeSecondaryNetworks
	ec2:DescribeSecondarySubnets
	ec2:DescribeSecurityGroupReferences
	ec2:DescribeSecurityGroupRules
	ec2:DescribeSecurityGroupVpcAssociations
	ec2:DescribeSecurityGroups
	ec2:DescribeServiceLinkVirtualInterfaces
	ec2:DescribeSnapshotAttribute
	ec2:DescribeSnapshotTierStatus
	ec2:DescribeSpotDatafeedSubscription
	ec2:DescribeSpotFleetInstances
	ec2:DescribeSpotFleetRequestHistory
	ec2:DescribeSpotFleetRequests
	ec2:DescribeSpotInstanceRequests
	ec2:DescribeSpotPriceHistory
	ec2:DescribeStaleSecurityGroups

Prefixo do serviço	Ações do
	ec2:DescribeStoreImageTasks
	ec2:DescribeTrafficMirrorFilterRules
	ec2:DescribeTrafficMirrorFilters
	ec2:DescribeTrafficMirrorSessions
	ec2:DescribeTrafficMirrorTargets
	ec2:DescribeTransitGatewayAttachments
	ec2:DescribeTransitGatewayConnectPeers
	ec2:DescribeTransitGatewayConnects
	ec2:DescribeTransitGatewayMeteringPolicies
	ec2:DescribeTransitGatewayMulticastDomains
	ec2:DescribeTransitGatewayPeeringAttachments
	ec2:DescribeTransitGatewayPolicyTables
	ec2:DescribeTransitGatewayRouteTableAnnouncements
	ec2:DescribeTransitGatewayRouteTables
	ec2:DescribeTransitGatewayVpcAttachments
	ec2:DescribeTransitGateways
	ec2:DescribeTrunkInterfaceAssociations
	ec2:DescribeVerifiedAccessEndpoints
	ec2:DescribeVerifiedAccessGroups
	ec2:DescribeVerifiedAccessInstanceLoggingConfigurations
	ec2:DescribeVerifiedAccessInstances

Prefixo do serviço	Ações do
	ec2:DescribeVerifiedAccessTrustProviders
	ec2:DescribeVolumeAttribute
	ec2:DescribeVolumeStatus
	ec2:DescribeVolumes
	ec2:DescribeVolumesModifications
	ec2:DescribeVpcAttribute
	ec2:DescribeVpcBlockPublicAccessExclusions
	ec2:DescribeVpcBlockPublicAccessOptions
	ec2:DescribeVpcClassicLink
	ec2:DescribeVpcClassicLinkDnsSupport
	ec2:DescribeVpcEncryptionControls
	ec2:DescribeVpcEndpointAssociations
	ec2:DescribeVpcEndpointConnectionNotifications
	ec2:DescribeVpcEndpointConnections
	ec2:DescribeVpcEndpointServiceConfigurations
	ec2:DescribeVpcEndpointServicePermissions
	ec2:DescribeVpcEndpointServices
	ec2:DescribeVpcEndpoints
	ec2:DescribeVpcPeeringConnections
	ec2:DescribeVpcs
	ec2:DescribeVpnConcentrators

Prefixo do serviço	Ações do
	ec2:DescribeVpnConnections
	ec2:DescribeVpnGateways
	ec2:DetachClassicLinkVpc
	ec2:DetachImageWatermark
	ec2:DetachInternetGateway
	ec2:DetachNetworkInterface
	ec2:DetachVerifiedAccessTrustProvider
	ec2:DetachVolume
	ec2:DetachVpnGateway
	ec2:DisableAddressTransfer
	ec2:DisableAllowedImagesSettings
	ec2:DisableAwsNetworkPerformanceMetricSubscription
	ec2:DisableCapacityManager
	ec2:DisableEbsEncryptionByDefault
	ec2:DisableFastLaunch
	ec2:DisableFastSnapshotRestores
	ec2:DisableImage
	ec2:DisableImageBlockPublicAccess
	ec2:DisableImageDeprecation
	ec2:DisableImageDeregistrationProtection
	ec2:DisableInstanceSqlHaStandbyDetections

Prefixo do serviço	Ações do
	ec2:DisableIpamOrganizationAdminAccount
	ec2:DisableIpamPolicy
	ec2:DisableRouteServerPropagation
	ec2:DisableSerialConsoleAccess
	ec2:DisableSnapshotBlockPublicAccess
	ec2:DisableTransitGatewayRouteTablePropagation
	ec2:DisableVgwRoutePropagation
	ec2:DisableVpcClassicLink
	ec2:DisableVpcClassicLinkDnsSupport
	ec2:DisassociateAddress
	ec2:DisassociateCapacityReservationBillingOwner
	ec2:DisassociateClientVpnTargetNetwork
	ec2:DisassociateEnclaveCertificateIamRole
	ec2:DisassociateIamInstanceProfile
	ec2:DisassociateInstanceEventWindow
	ec2:DisassociateIpamByoasn
	ec2:DisassociateIpamResourceDiscovery
	ec2:DisassociateNatGatewayAddress
	ec2:DisassociateRouteServer
	ec2:DisassociateRouteTable
	ec2:DisassociateSecurityGroupVpc

Prefixo do serviço	Ações do
	ec2:DisassociateSubnetCidrBlock
	ec2:DisassociateTransitGatewayMulticastDomain
	ec2:DisassociateTransitGatewayPolicyTable
	ec2:DisassociateTransitGatewayRouteTable
	ec2:DisassociateTrunkInterface
	ec2:DisassociateVpcCidrBlock
	ec2:EnableAddressTransfer
	ec2:EnableAllowedImagesSettings
	ec2:EnableAwsNetworkPerformanceMetricSubscription
	ec2:EnableCapacityManager
	ec2:EnableEbsEncryptionByDefault
	ec2:EnableFastLaunch
	ec2:EnableFastSnapshotRestores
	ec2:EnableImage
	ec2:EnableImageBlockPublicAccess
	ec2:EnableImageDeprecation
	ec2:EnableImageDeregistrationProtection
	ec2:EnableInstanceSqlHaStandbyDetections
	ec2:EnableIpamOrganizationAdminAccount
	ec2:EnableIpamPolicy
	ec2:EnableReachabilityAnalyzerOrganizationSharing

Prefixo do serviço	Ações do
	ec2:EnableRouteServerPropagation
	ec2:EnableSerialConsoleAccess
	ec2:EnableSnapshotBlockPublicAccess
	ec2:EnableTransitGatewayRouteTablePropagation
	ec2:EnableVgwRoutePropagation
	ec2:EnableVolumeIO
	ec2:EnableVpcClassicLink
	ec2:EnableVpcClassicLinkDnsSupport
	ec2:ExportClientVpnClientCertificateRevocationList
	ec2:ExportClientVpnClientConfiguration
	ec2:ExportImage
	ec2:ExportTransitGatewayRoutes
	ec2:ExportVerifiedAccessInstanceClientConfiguration
	ec2:GetActiveVpnTunnelStatus
	ec2:GetAllowedImagesSettings
	ec2:GetAssociatedEnclaveCertificateIamRoles
	ec2:GetAssociatedIpv6PoolCidrs
	ec2:GetAwsNetworkPerformanceData
	ec2:GetCapacityManagerAttributes
	ec2:GetCapacityManagerMetricData
	ec2:GetCapacityManagerMetricDimensions

Prefixo do serviço	Ações do
	ec2:GetCapacityReservationUsage
	ec2:GetCoipPoolUsage
	ec2:GetConsoleOutput
	ec2:GetConsoleScreenshot
	ec2:GetDeclarativePoliciesReportSummary
	ec2:GetDefaultCreditSpecification
	ec2:GetEbsDefaultKmsKeyId
	ec2:GetEbsEncryptionByDefault
	ec2:GetEnabledIpamPolicy
	ec2:GetFlowLogsIntegrationTemplate
	ec2:GetGroupsForCapacityReservation
	ec2:GetHostReservationPurchasePreview
	ec2:GetImageAncestry
	ec2:GetImageBlockPublicAccessState
	ec2:GetInstanceMetadataDefaults
	ec2:GetInstanceTpmEkPub
	ec2:GetInstanceTypesFromInstanceRequirements
	ec2:GetInstanceUefiData
	ec2:GetIpamAddressHistory
	ec2:GetIpamDiscoveredAccounts
	EC2: Obtenha endereços públicos descobertos pelo IPAM

Prefixo do serviço	Ações do
	ec2:GetIpamDiscoveredResourceCidrs
	ec2:GetIpamPolicyAllocationRules
	ec2:GetIpamPolicyOrganizationTargets
	ec2:GetIpamPoolAllocations
	ec2:GetIpamPoolCidrs
	ec2:GetIpamPrefixListResolverRules
	ec2:GetIpamPrefixListResolverVersionEntries
	ec2:GetIpamPrefixListResolverVersions
	ec2:GetIpamResourceCidrs
	ec2:GetLaunchTemplateData
	ec2:GetManagedPrefixListAssociations
	ec2:GetManagedPrefixListEntries
	ec2:GetManagedResourceVisibility
	ec2:GetNetworkInsightsAccessScopeAnalysisFindings
	ec2:GetNetworkInsightsAccessScopeContent
	ec2:GetPasswordData
	ec2:GetReservedInstancesExchangeQuote
	ec2:GetRouteServerAssociations
	ec2:GetRouteServerPropagations
	ec2:GetRouteServerRoutingDatabase
	ec2:GetSecurityGroupsForVpc

Prefixo do serviço	Ações do
	ec2:GetSerialConsoleAccessStatus
	ec2:GetSnapshotBlockPublicAccessState
	ec2:GetSpotPlacementScores
	ec2:GetSubnetCidrReservations
	ec2:GetTransitGatewayAttachmentPropagations
	ec2:GetTransitGatewayMeteringPolicyEntries
	ec2:GetTransitGatewayMulticastDomainAssociations
	ec2:GetTransitGatewayPolicyTableAssociations
	ec2:GetTransitGatewayPolicyTableEntries
	ec2:GetTransitGatewayPrefixListReferences
	ec2:GetTransitGatewayRouteTableAssociations
	ec2:GetTransitGatewayRouteTablePropagations
	ec2:GetVerifiedAccessEndpointPolicy
	ec2:GetVerifiedAccessEndpointTargets
	ec2:GetVerifiedAccessGroupPolicy
	ec2:GetVpcResourcesBlockingEncryptionEnforcement
	ec2:GetVpnConnectionDeviceSampleConfiguration
	ec2:GetVpnConnectionDeviceTypes
	ec2:GetVpnTunnelReplacementStatus
	ec2:ImportClientVpnClientCertificateRevocationList
	ec2:ImportImage

Prefixo do serviço	Ações do
	ec2:ImportInstance
	ec2:ImportKeyPair
	ec2:ImportSnapshot
	ec2:ImportVolume
	ec2:ListImagesInRecycleBin
	ec2:ListSnapshotsInRecycleBin
	ec2:ListVolumesInRecycleBin
	ec2:LockSnapshot
	ec2:ModifyAddressAttribute
	ec2:ModifyAvailabilityZoneGroup
	ec2:ModifyCapacityReservation
	ec2:ModifyCapacityReservationFleet
	ec2:ModifyClientVpnEndpoint
	ec2:ModifyDefaultCreditSpecification
	ec2:ModifyEbsDefaultKmsKeyId
	ec2:ModifyFleet
	ec2:ModifyFpgaImageAttribute
	ec2:ModifyHosts
	ec2:ModifyIdFormat
	ec2:ModifyIdentityIdFormat
	ec2:ModifyImageAttribute

Prefixo do serviço	Ações do
	ec2:ModifyInstanceAttribute
	ec2:ModifyInstanceCapacityReservationAttributes
	ec2:ModifyInstanceConnectEndpoint
	ec2:ModifyInstanceCpuOptions
	ec2:ModifyInstanceCreditSpecification
	ec2:ModifyInstanceEventStartTime
	ec2:ModifyInstanceEventWindow
	ec2:ModifyInstanceMaintenanceOptions
	ec2:ModifyInstanceMetadataDefaults
	ec2:ModifyInstanceMetadataOptions
	ec2:ModifyInstanceNetworkPerformanceOptions
	ec2:ModifyInstancePlacement
	ec2:ModifyIpam
	ec2:ModifyIpamPolicyAllocationRules
	ec2:ModifyIpamPool
	ec2:ModifyIpamPoolAllocation
	ec2:ModifyIpamPrefixListResolver
	ec2:ModifyIpamPrefixListResolverTarget
	ec2:ModifyIpamResourceCidr
	ec2:ModifyIpamResourceDiscovery
	ec2:ModifyIpamScope

Prefixo do serviço	Ações do
	ec2:ModifyLaunchTemplate
	ec2:ModifyLocalGatewayRoute
	ec2:ModifyManagedPrefixList
	ec2:ModifyManagedResourceVisibility
	ec2:ModifyNetworkInterfaceAttribute
	ec2:ModifyPrivateDnsNameOptions
	ec2:ModifyPublicIpDnsNameOptions
	ec2:ModifyReservedInstances
	ec2:ModifyRouteServer
	ec2:ModifySecurityGroupRules
	ec2:ModifySnapshotAttribute
	ec2:ModifySnapshotTier
	ec2:ModifySpotFleetRequest
	ec2:ModifySubnetAttribute
	ec2:ModifyTrafficMirrorFilterNetworkServices
	ec2:ModifyTrafficMirrorFilterRule
	ec2:ModifyTrafficMirrorSession
	ec2:ModifyTransitGateway
	ec2:ModifyTransitGatewayMeteringPolicy
	ec2:ModifyTransitGatewayPrefixListReference
	ec2:ModifyTransitGatewayVpcAttachment

Prefixo do serviço	Ações do
	ec2:ModifyVerifiedAccessEndpoint
	ec2:ModifyVerifiedAccessEndpointPolicy
	ec2:ModifyVerifiedAccessGroup
	ec2:ModifyVerifiedAccessGroupPolicy
	ec2:ModifyVerifiedAccessInstance
	ec2:ModifyVerifiedAccessInstanceLoggingConfiguration
	ec2:ModifyVerifiedAccessTrustProvider
	ec2:ModifyVolume
	ec2:ModifyVolumeAttribute
	ec2:ModifyVpcAttribute
	ec2:ModifyVpcBlockPublicAccessExclusion
	ec2:ModifyVpcBlockPublicAccessOptions
	ec2:ModifyVpcEncryptionControl
	ec2:ModifyVpcEndpoint
	ec2:ModifyVpcEndpointConnectionNotification
	ec2:ModifyVpcEndpointServiceConfiguration
	ec2:ModifyVpcEndpointServicePayerResponsibility
	ec2:ModifyVpcEndpointServicePermissions
	ec2:ModifyVpcPeeringConnectionOptions
	ec2:ModifyVpcTenancy
	ec2:ModifyVpnConnection

Prefixo do serviço	Ações do
	ec2:ModifyVpnConnectionOptions
	ec2:ModifyVpnTunnelCertificate
	ec2:ModifyVpnTunnelOptions
	ec2:MonitorInstances
	ec2:MoveAddressToVpc
	ec2:MoveByoipCidrToIpam
	ec2:MoveCapacityReservationInstances
	ec2:ProvisionByoipCidr
	ec2:ProvisionIpamByoasn
	ec2:ProvisionIpamPoolCidr
	ec2:ProvisionPublicIpv4PoolCidr
	ec2:PurchaseCapacityBlockExtension
	ec2:PurchaseHostReservation
	ec2:PurchaseReservedInstancesOffering
	ec2:PurchaseScheduledInstances
	ec2:RebootInstances
	ec2:RegisterImage
	ec2:RegisterInstanceEventNotificationAttributes
	ec2:RegisterTransitGatewayMulticastGroupMembers
	ec2:RegisterTransitGatewayMulticastGroupSources
	ec2:RejectCapacityReservationBillingOwnership

Prefixo do serviço	Ações do
	ec2:RejectTransitGatewayClientVpnAttachment
	ec2:RejectTransitGatewayMulticastDomainAssociations
	ec2:RejectTransitGatewayPeeringAttachment
	ec2:RejectTransitGatewayVpcAttachment
	ec2:RejectVpcEndpointConnections
	ec2:RejectVpcPeeringConnection
	ec2:ReleaseAddress
	ec2:ReleaseHosts
	ec2:ReleaseIpamPoolAllocation
	ec2:ReplaceIamInstanceProfileAssociation
	ec2:ReplaceImageCriteriaInAllowedImagesSettings
	ec2:ReplaceNetworkAclAssociation
	ec2:ReplaceNetworkAclEntry
	ec2:ReplaceRoute
	ec2:ReplaceRouteTableAssociation
	ec2:ReplaceTransitGatewayRoute
	ec2:ReplaceVpnTunnel
	ec2:ReportInstanceStatus
	ec2:RequestSpotFleet
	ec2:RequestSpotInstances
	ec2:ResetAddressAttribute

Prefixo do serviço	Ações do
	ec2:ResetEbsDefaultKmsKeyId
	ec2:ResetFpgaImageAttribute
	ec2:ResetImageAttribute
	ec2:ResetInstanceAttribute
	ec2:ResetNetworkInterfaceAttribute
	ec2:ResetSnapshotAttribute
	ec2:RestoreAddressToClassic
	ec2:RestoreImageFromRecycleBin
	ec2:RestoreManagedPrefixListVersion
	ec2:RestoreSnapshotFromRecycleBin
	ec2:RestoreSnapshotTier
	ec2:RestoreVolumeFromRecycleBin
	ec2:RevokeClientVpnIngress
	ec2:RevokeSecurityGroupEgress
	ec2:RevokeSecurityGroupIngress
	ec2:RunInstances
	ec2:RunScheduledInstances
	ec2:SearchLocalGatewayRoutes
	ec2:SearchTransitGatewayMulticastGroups
	ec2:SearchTransitGatewayRoutes
	ec2:SendDiagnosticInterrupt

Prefixo do serviço	Ações do
	ec2:StartDeclarativePoliciesReport
	ec2:StartInstances
	ec2:StartNetworkInsightsAccessScopeAnalysis
	ec2:StartNetworkInsightsAnalysis
	ec2:StartVpcEndpointServicePrivateDnsVerification
	ec2:TerminateClientVpnConnections
	ec2:UnassignIpv6Addresses
	ec2:UnassignPrivateIpAddresses
	ec2:UnassignPrivateNatGatewayAddress
	ec2:UnlockSnapshot
	ec2:UnmonitorInstances
	ec2:UpdateCapacityManagerOrganizationsAccess
	ec2:UpdateInterruptibleCapacityReservationAllocation
	ec2:UpdateSecurityGroupRuleDescriptionsEgress
	ec2:UpdateSecurityGroupRuleDescriptionsIngress
	ec2:WithdrawByoipCidr

Prefixo do serviço	Ações do
ecr	ecr:BatchCheckLayerAvailability ecr:BatchDeleteImage ecr:BatchGetImage ecr:BatchGetRepositoryScanningConfiguration ecr:CompleteLayerUpload ecr:CreatePullThroughCacheRule ecr:CreateRepositoryCreationTemplate ecr:DeleteLifecyclePolicy ecr:DeletePullThroughCacheRule ecr:DeleteRegistryPolicy ecr:DeleteRepository ecr:DeleteRepositoryCreationTemplate ecr:DeleteRepositoryPolicy ecr:DeleteSigningConfiguration ecr:DescribeImageReplicationStatus ecr:DescribeImageScanFindings ecr:DescribeImages ecr:DescribePullThroughCacheRules ecr:DescribeRegistry ecr:DescribeRepositories ecr:DescribeRepositoryCreationTemplates

Prefixo do serviço	Ações do
	ecr:GetAccountSetting ecr:GetAuthorizationToken ecr:GetDownloadUrlForLayer ecr:GetLifecyclePolicy ecr:GetLifecyclePolicyPreview ecr:GetRegistryPolicy ecr:GetRegistryScanningConfiguration ecr:GetRepositoryPolicy ecr:GetSigningConfiguration ecr:InitiateLayerUpload ecr:ListImages ecr:ListPullTimeUpdateExclusions ecr:PutAccountSetting ecr:PutImage ecr:PutImageScanningConfiguration ecr:PutRegistryPolicy ecr:PutRegistryScanningConfiguration ecr:PutReplicationConfiguration ecr:StartImageScan ecr:StartLifecyclePolicyPreview ecr:UpdatePullThroughCacheRule

Prefixo do serviço	Ações do
	ecr:UpdateRepositoryCreationTemplate ecr:UploadLayerPart ecr:ValidatePullThroughCacheRule

Prefixo do serviço	Ações do
ecr-public	ecr-public:BatchCheckLayerAvailability
	ecr-public:BatchDeleteImage
	ecr-public:CompleteLayerUpload
	ecr-public:CreateRepository
	ecr-public:DeleteRepository
	ecr-public:DeleteRepositoryPolicy
	ecr-public:DescribeImages
	ecr-public:DescribeRegistries
	ecr-public:DescribeRepositories
	ecr-public:GetAuthorizationToken
	ecr-public:GetRegistryCatalogData
	ecr-public:GetRepositoryCatalogData
	ecr-public:GetRepositoryPolicy
	ecr-public:InitiateLayerUpload
	ecr-public:PutImage
	ecr-public:PutRegistryCatalogData
	ecr-public:PutRepositoryCatalogData
	ecr-public:SetRepositoryPolicy
	ecr-public:UploadLayerPart

Prefixo do serviço	Ações do
ecs	ecs:ContinueServiceDeployment ecs:CreateCapacityProvider ecs:CreateCluster ecs:CreateDaemon ecs:CreateService ecs:CreateTaskSet ecs:DeleteAccountSetting ecs:DeleteAttributes ecs:DeleteCapacityProvider ecs:DeleteCluster ecs:DeleteDaemon ecs:DeleteDaemonTaskDefinition ecs:DeleteExpressGatewayService ecs:DeleteService ecs:DeleteTaskDefinitions ecs:DeleteTaskSet ecs:DeregisterContainerInstance ecs:DeregisterTaskDefinition ecs:DescribeCapacityProviders ecs:DescribeClusters ecs:DescribeContainerInstances

Prefixo do serviço	Ações do
	ecs:DescribeDaemon ecs:DescribeDaemonDeployments ecs:DescribeDaemonRevisions ecs:DescribeDaemonTaskDefinition ecs:DescribeExpressGatewayService ecs:DescribeServiceDeployments ecs:DescribeServiceRevisions ecs:DescribeServices ecs:DescribeTaskDefinition ecs:DescribeTaskSets ecs:DescribeTasks ecs:DiscoverPollEndpoint ecs:ExecuteCommand ecs:GetTaskProtection ecs:ListAccountSettings ecs:ListAttributes ecs:ListClusters ecs:ListContainerInstances ecs:ListDaemonDeployments ecs:ListDaemonTaskDefinitions ecs:ListDaemons

Prefixo do serviço	Ações do
	ecs:ListServiceDeployments ecs:ListServices ecs:ListServicesByNamespace ecs:ListTaskDefinitionFamilies ecs:ListTaskDefinitions ecs:ListTasks ecs:PutAccountSetting ecs:PutAccountSettingDefault ecs:PutAttributes ecs:PutClusterCapacityProviders ecs:RegisterContainerInstance ecs:RegisterDaemonTaskDefinition ecs:RunTask ecs:StartTask ecs:StopServiceDeployment ecs:StopTask ecs:SubmitAttachmentStateChanges ecs:SubmitContainerStateChange ecs:SubmitTaskStateChange ecs:UpdateCapacityProvider ecs:UpdateCluster

Prefixo do serviço	Ações do
	ecs:UpdateClusterSettings ecs:UpdateContainerAgent ecs:UpdateContainerInstancesState ecs:UpdateDaemon ecs:UpdateExpressGatewayService ecs:UpdateService ecs:UpdateServicePrimaryTaskSet ecs:UpdateTaskProtection ecs:UpdateTaskSet

Prefixo do serviço	Ações do
eks	eks:AssociateAccessPolicy
	eks:AssociateEncryptionConfig
	eks:AssociateIdentityProviderConfig
	eks:CancelUpdate
	eks:CreateAccessEntry
	eks:CreateAddon
	eks:CreateCluster
	eks:CreateEksAnywhereSubscription
	eks:CreateFargateProfile
	eks:CreateNodegroup
	eks:DeleteAccessEntry
	eks:DeleteAddon
	eks:DeleteCapability
	eks:DeleteCluster
	eks:DeleteEksAnywhereSubscription
	eks:DeleteFargateProfile
	eks:DeleteNodegroup
	eks:DeletePodIdentityAssociation
	eks:DeregisterCluster
	eks:DescribeAccessEntry
	eks:DescribeAddon

Prefixo do serviço	Ações do
	eks:DescribeAddonConfiguration
	eks:DescribeAddonVersions
	eks:DescribeCapability
	eks:DescribeCluster
	eks:DescribeClusterVersions
	eks:DescribeEksAnywhereSubscription
	eks:DescribeFargateProfile
	eks:DescribeIdentityProviderConfig
	eks:DescribeInsight
	eks:DescribeInsightsRefresh
	eks:DescribeNodegroup
	eks:DescribePodIdentityAssociation
	eks:DescribeUpdate
	eks:DisassociateAccessPolicy
	eks:DisassociateIdentityProviderConfig
	eks:ListAccessEntries
	eks:ListAccessPolicies
	eks:ListAddons
	eks:ListAssociatedAccessPolicies
	eks:ListCapabilities
	eks:ListClusters

Prefixo do serviço	Ações do
	eks:ListEksAnywhereSubscriptions
	eks:ListFargateProfiles
	eks:ListIdentityProviderConfigs
	eks:ListInsights
	eks:ListNodegroups
	eks:ListPodIdentityAssociations
	eks:ListUpdates
	eks:RegisterCluster
	eks:StartInsightsRefresh
	eks:UpdateAccessEntry
	eks:UpdateAddon
	eks:UpdateCapability
	eks:UpdateClusterConfig
	eks:UpdateClusterVersion
	eks:UpdateEksAnywhereSubscription
	eks:UpdateNodegroupConfig
	eks:UpdateNodegroupVersion
	eks:UpdatePodIdentityAssociation

Prefixo do serviço	Ações do
elasticache	elasticache:AuthorizeCacheSecurityGroupIngress elasticache:BatchApplyUpdateAction elasticache:BatchStopUpdateAction elasticache:CompleteMigration elasticache:CopyServerlessCacheSnapshot elasticache:CopySnapshot elasticache:CreateCacheCluster elasticache:CreateCacheParameterGroup elasticache:CreateCacheSecurityGroup elasticache:CreateCacheSubnetGroup elasticache:CreateGlobalReplicationGroup elasticache:CreateReplicationGroup elasticache:CreateServerlessCache elasticache:CreateServerlessCacheSnapshot elasticache:CreateSnapshot elasticache:CreateUser elasticache:CreateUserGroup elasticache:DecreaseNodeGroupsInGlobalReplicationGroup elasticache:DecreaseReplicaCount elasticache:DeleteCacheCluster elasticache:DeleteCacheParameterGroup

Prefixo do serviço	Ações do
	elasticache:DeleteCacheSecurityGroup elasticache:DeleteCacheSubnetGroup elasticache:DeleteGlobalReplicationGroup elasticache:DeleteReplicationGroup elasticache:DeleteServerlessCache elasticache:DeleteServerlessCacheSnapshot elasticache:DeleteSnapshot elasticache:DeleteUser elasticache:DeleteUserGroup elasticache:DescribeCacheClusters elasticache:DescribeCacheEngineVersions elasticache:DescribeCacheParameterGroups elasticache:DescribeCacheParameters elasticache:DescribeCacheSecurityGroups elasticache:DescribeCacheSubnetGroups elasticache:DescribeEngineDefaultParameters elasticache:DescribeEvents elasticache:DescribeGlobalReplicationGroups elasticache:DescribeReplicationGroups elasticache:DescribeReservedCacheNodes elasticache:DescribeReservedCacheNodesOfferings

Prefixo do serviço	Ações do
	elasticache:DescribeServerlessCacheSnapshots elasticache:DescribeServerlessCaches elasticache:DescribeServiceUpdates elasticache:DescribeSnapshots elasticache:DescribeUpdateActions elasticache:DescribeUserGroups elasticache:DescribeUsers elasticache:DisassociateGlobalReplicationGroup elasticache:ExportServerlessCacheSnapshot elasticache:FailoverGlobalReplicationGroup elasticache:IncreaseNodeGroupsInGlobalReplicationGroup elasticache:IncreaseReplicaCount elasticache:ListAllowedNodeTypeModifications elasticache:ModifyCacheCluster elasticache:ModifyCacheParameterGroup elasticache:ModifyCacheSubnetGroup elasticache:ModifyGlobalReplicationGroup elasticache:ModifyReplicationGroup elasticache:ModifyReplicationGroupShardConfiguration elasticache:ModifyServerlessCache elasticache:ModifyUser

Prefixo do serviço	Ações do
	elasticache:ModifyUserGroup elasticache:PurchaseReservedCacheNodesOffering elasticache:RebalanceSlotsInGlobalReplicationGroup elasticache:RebootCacheCluster elasticache:ResetCacheParameterGroup elasticache:RevokeCacheSecurityGroupIngress elasticache:StartMigration elasticache:TestFailover elasticache:TestMigration

Prefixo do serviço	Ações do
elasticbeanstalk	elasticbeanstalk:AbortEnvironmentUpdate
	elasticbeanstalk:ApplyEnvironmentManagedAction
	elasticbeanstalk:AssociateEnvironmentOperationsRole
	elasticbeanstalk:CheckDNSAvailability
	elasticbeanstalk:ComposeEnvironments
	elasticbeanstalk:CreateApplication
	elasticbeanstalk:CreateApplicationVersion
	elasticbeanstalk:CreateConfigurationTemplate
	elasticbeanstalk:CreateEnvironment
	elasticbeanstalk:CreatePlatformVersion
	elasticbeanstalk:CreateStorageLocation
	elasticbeanstalk:DeleteApplication
	elasticbeanstalk:DeleteApplicationVersion
	elasticbeanstalk:DeleteConfigurationTemplate
	elasticbeanstalk:DeleteEnvironmentConfiguration
	elasticbeanstalk:DeletePlatformVersion
	elasticbeanstalk:DescribeAccountAttributes
	elasticbeanstalk:DescribeApplicationVersions
	elasticbeanstalk:DescribeApplications
	elasticbeanstalk:DescribeConfigurationOptions
	elasticbeanstalk:DescribeConfigurationSettings

Prefixo do serviço	Ações do
	elasticbeanstalk:DescribeEnvironmentHealth
	elasticbeanstalk:DescribeEnvironmentManagedActionHistory
	elasticbeanstalk:DescribeEnvironmentManagedActions
	elasticbeanstalk:DescribeEnvironmentResources
	elasticbeanstalk:DescribeEnvironments
	elasticbeanstalk:DescribeEvents
	elasticbeanstalk:DescribeInstancesHealth
	elasticbeanstalk:DescribePlatformVersion
	elasticbeanstalk:DisassociateEnvironmentOperationsRole
	elasticbeanstalk:ListAvailableSolutionStacks
	elasticbeanstalk:ListPlatformBranches
	elasticbeanstalk:ListPlatformVersions
	elasticbeanstalk:RebuildEnvironment
	elasticbeanstalk:RequestEnvironmentInfo
	elasticbeanstalk:RestartAppServer
	elasticbeanstalk:RetrieveEnvironmentInfo
	elasticbeanstalk:SwapEnvironmentCNAMEs
	elasticbeanstalk:TerminateEnvironment
	elasticbeanstalk:UpdateApplication
	elasticbeanstalk:UpdateApplicationResourceLifecycle
	elasticbeanstalk:UpdateApplicationVersion

Prefixo do serviço	Ações do
	elasticbeanstalk:UpdateConfigurationTemplate elasticbeanstalk:UpdateEnvironment elasticbeanstalk:ValidateConfigurationSettings

Prefixo do serviço	Ações do
elasticfilesystem	elasticfilesystem:CreateAccessPoint elasticfilesystem:CreateFileSystem elasticfilesystem:CreateMountTarget elasticfilesystem:CreateReplicationConfiguration elasticfilesystem:DeleteAccessPoint elasticfilesystem:DeleteFileSystem elasticfilesystem:DeleteFileSystemPolicy elasticfilesystem:DeleteMountTarget elasticfilesystem:DeleteReplicationConfiguration elasticfilesystem:DescribeAccessPoints elasticfilesystem:DescribeAccountPreferences elasticfilesystem:DescribeBackupPolicy elasticfilesystem:DescribeFileSystemPolicy elasticfilesystem:DescribeFileSystems elasticfilesystem:DescribeLifecycleConfiguration elasticfilesystem:DescribeMountTargetSecurityGroups elasticfilesystem:DescribeMountTargets elasticfilesystem:DescribeReplicationConfigurations elasticfilesystem:ModifyMountTargetSecurityGroups elasticfilesystem:PutAccountPreferences elasticfilesystem:PutBackupPolicy

Prefixo do serviço	Ações do
	elasticfilesystem:PutFileSystemPolicy elasticfilesystem:PutLifecycleConfiguration elasticfilesystem:UpdateFileSystem elasticfilesystem:UpdateFileSystemProtection

Prefixo do serviço	Ações do
elasticloadbalancing	elasticloadbalancing:AddListenerCertificates
	elasticloadbalancing:AddTrustStoreRevocations
	elasticloadbalancing:ApplySecurityGroupsToLoadBalancer
	elasticloadbalancing:AttachLoadBalancerToSubnets
	elasticloadbalancing:ConfigureHealthCheck
	elasticloadbalancing:CreateAppCookieStickinessPolicy
	elasticloadbalancing:CreateLBCookieStickinessPolicy
	elasticloadbalancing:CreateListener
	elasticloadbalancing:CreateLoadBalancer
	elasticloadbalancing:CreateLoadBalancerListeners
	elasticloadbalancing:CreateLoadBalancerPolicy
	elasticloadbalancing:CreateRule
	elasticloadbalancing:CreateTargetGroup
	elasticloadbalancing:CreateTrustStore
	elasticloadbalancing:DeleteListener
	elasticloadbalancing:DeleteLoadBalancer
	elasticloadbalancing:DeleteLoadBalancerListeners
	elasticloadbalancing:DeleteLoadBalancerPolicy
	elasticloadbalancing:DeleteRule
	elasticloadbalancing:DeleteSharedTrustStoreAssociation
	elasticloadbalancing:DeleteTargetGroup

Prefixo do serviço	Ações do
	elasticloadbalancing:DeleteTrustStore
	elasticloadbalancing:DeregisterInstancesFromLoadBalancer
	elasticloadbalancing:DeregisterTargets
	elasticloadbalancing:DescribeAccountLimits
	elasticloadbalancing:DescribeCapacityReservation
	elasticloadbalancing:DescribeInstanceHealth
	elasticloadbalancing:DescribeListenerAttributes
	elasticloadbalancing:DescribeListenerCertificates
	elasticloadbalancing:DescribeListeners
	elasticloadbalancing:DescribeLoadBalancerAttributes
	elasticloadbalancing:DescribeLoadBalancerPolicies
	elasticloadbalancing:DescribeLoadBalancerPolicyTypes
	elasticloadbalancing:DescribeLoadBalancers
	elasticloadbalancing:DescribeRules
	elasticloadbalancing:DescribeSSLPolicies
	elasticloadbalancing:DescribeTargetGroupAttributes
	elasticloadbalancing:DescribeTargetGroups
	elasticloadbalancing:DescribeTargetHealth
	elasticloadbalancing:DescribeTrustStoreAssociations
	elasticloadbalancing:DescribeTrustStoreRevocations
	elasticloadbalancing:DescribeTrustStores

Prefixo do serviço	Ações do
	elasticloadbalancing:DetachLoadBalancerFromSubnets
	elasticloadbalancing:DisableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:EnableAvailabilityZonesForLoadBalancer
	elasticloadbalancing:GetResourcePolicy
	elasticloadbalancing:GetTrustStoreCaCertificatesBundle
	elasticloadbalancing:GetTrustStoreRevocationContent
	elasticloadbalancing:ModifyCapacityReservation
	elasticloadbalancing:ModifyIpPools
	elasticloadbalancing:ModifyListener
	elasticloadbalancing:ModifyLoadBalancerAttributes
	elasticloadbalancing:ModifyRule
	elasticloadbalancing:ModifyTargetGroup
	elasticloadbalancing:ModifyTargetGroupAttributes
	elasticloadbalancing:ModifyTrustStore
	elasticloadbalancing:RegisterInstancesWithLoadBalancer
	elasticloadbalancing:RegisterTargets
	elasticloadbalancing:RemoveListenerCertificates
	elasticloadbalancing:RemoveTrustStoreRevocations
	elasticloadbalancing:SetIpAddressType
	elasticloadbalancing:SetLoadBalancerListenerSSLCertificate
	elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer

Prefixo do serviço	Ações do
	elasticloadbalancing:SetLoadBalancerPoliciesOfListener
	elasticloadbalancing:SetRulePriorities
	elasticloadbalancing:SetSecurityGroups
	elasticloadbalancing:SetSubnets

Prefixo do serviço	Ações do
emr-containers	emr-containers:CancelJobRun emr-containers:CreateJobTemplate emr-containers:CreateManagedEndpoint emr-containers:CreateSecurityConfiguration emr-containers:CreateVirtualCluster emr-containers:DeleteJobTemplate emr-containers:DeleteManagedEndpoint emr-containers:DeleteSecurityConfiguration emr-containers:DeleteVirtualCluster emr-containers:DescribeJobRun emr-containers:DescribeJobTemplate emr-containers:DescribeManagedEndpoint emr-containers:DescribeSecurityConfiguration emr-containers:DescribeVirtualCluster emr-containers:GetManagedEndpointSessionCredentials emr-containers:ListJobRuns emr-containers:ListJobTemplates emr-containers:ListManagedEndpoints emr-containers:ListSecurityConfigurations emr-containers:ListVirtualClusters emr-containers:StartJobRun

Prefixo do serviço	Ações do
emr-serverless	emr-serverless:CancelJobRun
	emr-serverless:CreateApplication
	emr-serverless:DeleteApplication
	emr-serverless:GetApplication
	emr-serverless:GetDashboardForJobRun
	emr-serverless:GetJobRun
	emr-serverless:GetResourceDashboard
	emr-serverless:GetSession
	emr-serverless:GetSessionEndpoint
	emr-serverless:ListApplications
	emr-serverless:ListJobRunAttempts
	emr-serverless:ListJobRuns
	emr-serverless:ListSessions
	emr-serverless:StartApplication
	emr-serverless:StartJobRun
	emr-serverless:StopApplication
	emr-serverless:TerminateSession
	emr-serverless:UpdateApplication

Prefixo do serviço	Ações do
es	es:AcceptInboundConnection es:AcceptInboundCrossClusterSearchConnection es:AssociatePackage es:AuthorizeVpcEndpointAccess es:CancelElasticsearchServiceSoftwareUpdate es:CancelServiceSoftwareUpdate es:CreateDomain es:CreateElasticsearchDomain es:CreateIndex es:CreateOutboundConnection es:CreateOutboundCrossClusterSearchConnection es:CreatePackage es:CreateVpcEndpoint es:DeleteDomain es:DeleteElasticsearchDomain es:DeleteElasticsearchServiceRole es:DeleteInboundConnection es:DeleteInboundCrossClusterSearchConnection es:DeleteIndex es:DeleteOutboundConnection es:DeleteOutboundCrossClusterSearchConnection

Prefixo do serviço	Ações do
	es:DeletePackage es:DeleteVpcEndpoint es:DescribeDomain es:DescribeDomainAutoTunes es:DescribeDomainChangeProgress es:DescribeDomainConfig es:DescribeDomainHealth es:DescribeDomainNodes es:DescribeDomains es:DescribeDryRunProgress es:DescribeElasticsearchDomain es:DescribeElasticsearchDomainConfig es:DescribeElasticsearchDomains es:DescribeElasticsearchInstanceTypeLimits es:DescribeInboundConnections es:DescribeInboundCrossClusterSearchConnections es:DescribeInstanceTypeLimits es:DescribeOutboundConnections es:DescribeOutboundCrossClusterSearchConnections es:DescribePackages es:DescribeReservedElasticsearchInstanceOfferings

Prefixo do serviço	Ações do
	es:DescribeReservedElasticsearchInstances es:DescribeReservedInstanceOfferings es:DescribeReservedInstances es:DescribeVpcEndpoints es:DissociatePackage es:DissociatePackages es:GetCompatibleElasticsearchVersions es:GetCompatibleVersions es:GetDataSource es:GetDomainMaintenanceStatus es:GetPackageVersionHistory es:GetUpgradeHistory es:GetUpgradeStatus es:ListDataSources es:ListDomainNames es:ListDomainsForPackage es:ListElasticsearchInstanceTypes es:ListElasticsearchVersions es:ListInsights es:ListInstanceTypeDetails es:ListPackagesForDomain

Prefixo do serviço	Ações do
	es:ListScheduledActions es:ListVersions es:ListVpcEndpointAccess es:ListVpcEndpoints es:ListVpcEndpointsForDomain es:PurchaseReservedElasticsearchInstanceOffering es:PurchaseReservedInstanceOffering es:RejectInboundConnection es:RejectInboundCrossClusterSearchConnection es:RevokeVpcEndpointAccess es:RollbackServiceSoftwareUpdate es:StartDomainMaintenance es:StartElasticsearchServiceSoftwareUpdate es:StartServiceSoftwareUpdate es:UpdateDataSource es:UpdateDomainConfig es:UpdateElasticsearchDomainConfig es:UpdateIndex es:UpdatePackage es:UpdatePackageScope es:UpdateScheduledAction

Prefixo do serviço	Ações do
	es:UpdateVpcEndpoint es:UpgradeDomain es:UpgradeElasticsearchDomain

Prefixo do serviço	Ações do
eventos	events:ActivateEventSource
	events:CancelReplay
	events:CreateApiDestination
	events:CreateArchive
	events:CreateConnection
	events:CreateEndpoint
	events:CreateEventBus
	events:CreatePartnerEventSource
	events:DeactivateEventSource
	events:DeauthorizeConnection
	events:DeleteApiDestination
	events:DeleteArchive
	events:DeleteConnection
	events:DeleteEndpoint
	events:DeleteEventBus
	events:DeletePartnerEventSource
	events:DeleteRule
	events:DescribeApiDestination
	events:DescribeArchive
	events:DescribeConnection
	events:DescribeEndpoint

Prefixo do serviço	Ações do
	events:DescribeEventBus
	events:DescribeEventSource
	events:DescribePartnerEventSource
	events:DescribeReplay
	events:DescribeRule
	events:DisableRule
	events:EnableRule
	events:ListApiDestinations
	events:ListArchives
	events:ListConnections
	events:ListEndpoints
	events:ListEventBuses
	events:ListEventSources
	events:ListPartnerEventSourceAccounts
	events:ListPartnerEventSources
	events:ListReplays
	events:ListRuleNamesByTarget
	events:ListRules
	events:ListTargetsByRule
	events:PutPermission
	events:PutRule

Prefixo do serviço	Ações do
	events:PutTargets
	events:RemovePermission
	events:RemoveTargets
	events:StartReplay
	events:TestEventPattern
	events:UpdateApiDestination
	events:UpdateArchive
	events:UpdateConnection
	events:UpdateEndpoint
	events:UpdateEventBus

Prefixo do serviço	Ações do
finspace	finspace:CreateEnvironment finspace:CreateKxChangeset finspace:CreateKxCluster finspace:CreateKxDatabase finspace:CreateKxDataview finspace:CreateKxEnvironment finspace:CreateKxScalingGroup finspace:CreateKxUser finspace:CreateKxVolume finspace:CreateUser finspace:DeleteEnvironment finspace:DeleteKxCluster finspace:DeleteKxClusterNode finspace:DeleteKxDatabase finspace:DeleteKxDataview finspace:DeleteKxEnvironment finspace:DeleteKxScalingGroup finspace:DeleteKxUser finspace:DeleteKxVolume finspace:GetEnvironment finspace:GetKxChangeset

Prefixo do serviço	Ações do
	finspace:GetKxCluster finspace:GetKxConnectionString finspace:GetKxDatabase finspace:GetKxDataview finspace:GetKxEnvironment finspace:GetKxScalingGroup finspace:GetKxUser finspace:GetKxVolume finspace:GetLoadSampleDataSetGroupIntoEnvironmentStatus finspace:GetUser finspace:ListEnvironments finspace:ListKxChangesets finspace:ListKxClusterNodes finspace:ListKxClusters finspace:ListKxDatabases finspace:ListKxDataviews finspace:ListKxEnvironments finspace:ListKxScalingGroups finspace:ListKxUsers finspace:ListKxVolumes finspace:ListUsers

Prefixo do serviço	Ações do
	finspace:LoadSampleDataSetGroupIntoEnvironment finspace:ResetUserPassword finspace:UpdateEnvironment finspace:UpdateKxClusterCodeConfiguration finspace:UpdateKxClusterDatabases finspace:UpdateKxDatabase finspace:UpdateKxDataview finspace:UpdateKxEnvironment finspace:UpdateKxEnvironmentNetwork finspace:UpdateKxUser finspace:UpdateKxVolume finspace:UpdateUser
firehose	firehose:CreateDeliveryStream firehose:DeleteDeliveryStream firehose:DescribeDeliveryStream firehose:ListDeliveryStreams firehose:StartDeliveryStreamEncryption firehose:StopDeliveryStreamEncryption firehose:UpdateDestination

Prefixo do serviço	Ações do
fis	fis:CreateExperimentTemplate fis:CreateTargetAccountConfiguration fis:DeleteExperimentTemplate fis:DeleteTargetAccountConfiguration fis:GetAction fis:GetExperiment fis:GetExperimentTargetAccountConfiguration fis:GetExperimentTemplate fis:GetSafetyLever fis:GetTargetAccountConfiguration fis:GetTargetResourceType fis:ListActions fis:ListExperimentResolvedTargets fis:ListExperimentTargetAccountConfigurations fis:ListExperimentTemplates fis:ListExperiments fis:ListTargetAccountConfigurations fis:ListTargetResourceTypes fis:StartExperiment fis:StopExperiment fis:UpdateExperimentTemplate

Prefixo do serviço	Ações do
	fis:UpdateSafetyLeverState fis:UpdateTargetAccountConfiguration

Prefixo do serviço	Ações do
fms	fms:AssociateAdminAccount
	fms:AssociateThirdPartyFirewall
	fms:BatchAssociateResource
	fms:BatchDisassociateResource
	fms:DeleteAppsList
	fms:DeleteNotificationChannel
	fms:DeletePolicy
	fms:DeleteProtocolsList
	fms:DeleteResourceSet
	fms:DisassociateAdminAccount
	fms:DisassociateThirdPartyFirewall
	fms:GetAdminAccount
	fms:GetAdminScope
	fms:GetAppsList
	fms:GetComplianceDetail
	fms:GetNotificationChannel
	fms:GetPolicy
	fms:GetProtectionStatus
	fms:GetProtocolsList
	fms:GetResourceSet
	fms:GetThirdPartyFirewallAssociationStatus

Prefixo do serviço	Ações do
	fms:GetViolationDetails
	fms:ListAdminAccountsForOrganization
	fms:ListAdminsManagingAccount
	fms:ListAppsLists
	fms:ListComplianceStatus
	fms:ListDiscoveredResources
	fms:ListMemberAccounts
	fms:ListPolicies
	fms:ListProtocolsLists
	fms:ListResourceSetResources
	fms:ListResourceSets
	fms:ListThirdPartyFirewallFirewallPolicies
	fms:PutAdminAccount
	fms:PutAppsList
	fms:PutNotificationChannel
	fms:PutPolicy
	fms:PutProtocolsList
	fms:PutResourceSet

Prefixo do serviço	Ações do
frauddetector	frauddetector:BatchCreateVariable frauddetector:BatchGetVariable frauddetector:CancelBatchImportJob frauddetector:CancelBatchPredictionJob frauddetector:CreateBatchImportJob frauddetector:CreateBatchPredictionJob frauddetector:CreateDetectorVersion frauddetector:CreateList frauddetector:CreateModel frauddetector:CreateModelVersion frauddetector:CreateRule frauddetector:CreateVariable frauddetector:DeleteBatchImportJob frauddetector:DeleteBatchPredictionJob frauddetector:DeleteDetector frauddetector:DeleteDetectorVersion frauddetector:DeleteEntityType frauddetector:DeleteEvent frauddetector:DeleteEventType frauddetector:DeleteEventsByEventType frauddetector:DeleteExternalModel

Prefixo do serviço	Ações do
	frauddetector:DeleteLabel frauddetector:DeleteList frauddetector:DeleteModel frauddetector:DeleteModelVersion frauddetector:DeleteOutcome frauddetector:DeleteRule frauddetector:DeleteVariable frauddetector:DescribeDetector frauddetector:DescribeModelVersions frauddetector:GetBatchImportJobs frauddetector:GetBatchPredictionJobs frauddetector:GetDeleteEventsByEventTypeStatus frauddetector:GetDetectorVersion frauddetector:GetDetectors frauddetector:GetEntityTypes frauddetector:GetEvent frauddetector:GetEventPrediction frauddetector:GetEventPredictionMetadata frauddetector:GetEventTypes frauddetector:GetExternalModels frauddetector:GetKMSEncryptionKey

Prefixo do serviço	Ações do
	frauddetector:GetLabels frauddetector:GetListElements frauddetector:GetListsMetadata frauddetector:GetModelVersion frauddetector:GetModels frauddetector:GetOutcomes frauddetector:GetRules frauddetector:GetVariables frauddetector:ListEventPredictions frauddetector:PutDetector frauddetector:PutEntityType frauddetector:PutEventType frauddetector:PutExternalModel frauddetector:PutKMSEncryptionKey frauddetector:PutLabel frauddetector:PutOutcome frauddetector:SendEvent frauddetector:UpdateDetectorVersion frauddetector:UpdateDetectorVersionMetadata frauddetector:UpdateDetectorVersionStatus frauddetector:UpdateEventLabel

Prefixo do serviço	Ações do
	frauddetector:UpdateList frauddetector:UpdateModel frauddetector:UpdateModelVersion frauddetector:UpdateModelVersionStatus frauddetector:UpdateRuleMetadata frauddetector:UpdateRuleVersion frauddetector:UpdateVariable

Prefixo do serviço	Ações do
fsx	fsx:AssociateFileSystemAliases
	fsx:CancelDataRepositoryTask
	fsx:CopyBackup
	fsx:CreateDataRepositoryTask
	fsx:CreateFileCache
	fsx:CreateFileSystem
	fsx:CreateFileSystemFromBackup
	fsx:CreateSnapshot
	fsx:CreateStorageVirtualMachine
	fsx:CreateVolume
	fsx:CreateVolumeFromBackup
	fsx:DeleteBackup
	fsx:DeleteFileCache
	fsx:DeleteFileSystem
	fsx:DeleteSnapshot
	fsx:DeleteStorageVirtualMachine
	fsx:DeleteVolume
	fsx:DescribeBackups
	fsx:DescribeDataRepositoryAssociations
	fsx:DescribeDataRepositoryTasks
	fsx:DescribeFileCaches

Prefixo do serviço	Ações do
	fsx:DescribeFileSystemAliases
	fsx:DescribeFileSystems
	fsx:DescribeS3AccessPointAttachments
	fsx:DescribeSharedVpcConfiguration
	fsx:DescribeSnapshots
	fsx:DescribeStorageVirtualMachines
	fsx:DescribeVolumes
	fsx:DetachAndDeleteS3AccessPoint
	fsx:DisassociateFileSystemAliases
	fsx:ReleaseFileSystemNfsV3Locks
	fsx:RestoreVolumeFromSnapshot
	fsx:StartMisconfiguredStateRecovery
	fsx:UpdateDataRepositoryAssociation
	fsx:UpdateFileCache
	fsx:UpdateFileSystem
	fsx:UpdateSharedVpcConfiguration
	fsx:UpdateSnapshot
	fsx:UpdateStorageVirtualMachine
	fsx:UpdateVolume

Prefixo do serviço	Ações do
gamelift	gamelift:AcceptMatch gamelift:ClaimGameServer gamelift:CreateAlias gamelift:CreateBuild gamelift:CreateContainerGroupDefinition gamelift:CreateFleet gamelift:CreateFleetLocations gamelift:CreateGameServerGroup gamelift:CreateGameSession gamelift:CreateGameSessionQueue gamelift:CreateLocation gamelift:CreateMatchmakingConfiguration gamelift:CreateMatchmakingRuleSet gamelift:CreatePlayerSession gamelift:CreatePlayerSessions gamelift:CreateScript gamelift:CreateVpcPeeringAuthorization gamelift:CreateVpcPeeringConnection gamelift:DeleteAlias gamelift:DeleteBuild gamelift:DeleteContainerGroupDefinition

Prefixo do serviço	Ações do
	gamelift:DeleteFleet gamelift:DeleteFleetLocations gamelift:DeleteGameServerGroup gamelift:DeleteGameSessionQueue gamelift:DeleteLocation gamelift:DeleteMatchmakingConfiguration gamelift:DeleteMatchmakingRuleSet gamelift:DeleteScalingPolicy gamelift:DeleteScript gamelift:DeleteVpcPeeringAuthorization gamelift:DeleteVpcPeeringConnection gamelift:DeregisterCompute gamelift:DeregisterGameServer gamelift:DescribeAlias gamelift:DescribeBuild gamelift:DescribeCompute gamelift:DescribeContainerFleet gamelift:DescribeContainerGroupDefinition gamelift:DescribeContainerGroupPortMappings gamelift:DescribeEC2InstanceLimits gamelift:DescribeFleetAttributes

Prefixo do serviço	Ações do
	gamelift:DescribeFleetCapacity gamelift:DescribeFleetEvents gamelift:DescribeFleetLocationAttributes gamelift:DescribeFleetLocationCapacity gamelift:DescribeFleetLocationUtilization gamelift:DescribeFleetPortSettings gamelift:DescribeFleetUtilization gamelift:DescribeGameServer gamelift:DescribeGameServerGroup gamelift:DescribeGameServerInstances gamelift:DescribeGameSessionDetails gamelift:DescribeGameSessionPlacement gamelift:DescribeGameSessionQueues gamelift:DescribeGameSessions gamelift:DescribeInstances gamelift:DescribeMatchmaking gamelift:DescribeMatchmakingConfigurations gamelift:DescribeMatchmakingRuleSets gamelift:DescribePlayerSessions gamelift:DescribeRuntimeConfiguration gamelift:DescribeScalingPolicies

Prefixo do serviço	Ações do
	gamelift:DescribeScript gamelift:DescribeVpcPeeringAuthorizations gamelift:DescribeVpcPeeringConnections gamelift:GetComputeAccess gamelift:GetComputeAuthToken gamelift:GetGameSessionLogUrl gamelift:GetInstanceAccess gamelift:GetPlayerConnectionDetails gamelift:ListAliases gamelift:ListBuilds gamelift:ListCompute gamelift:ListContainerFleets gamelift:ListContainerGroupDefinitionVersions gamelift:ListContainerGroupDefinitions gamelift:ListFleetDeployments gamelift:ListFleets gamelift:ListGameServerGroups gamelift:ListGameServers gamelift:ListLocations gamelift:ListScripts gamelift:PutScalingPolicy

Prefixo do serviço	Ações do
	gamelift:RegisterCompute gamelift:RegisterGameServer gamelift:RequestUploadCredentials gamelift:ResolveAlias gamelift:ResumeGameServerGroup gamelift:SearchGameSessions gamelift:StartFleetActions gamelift:StartGameSessionPlacement gamelift:StartMatchBackfill gamelift:StartMatchmaking gamelift:StopFleetActions gamelift:StopGameSessionPlacement gamelift:StopMatchmaking gamelift:SuspendGameServerGroup gamelift:TerminateGameSession gamelift:UpdateAlias gamelift:UpdateBuild gamelift:UpdateContainerGroupDefinition gamelift:UpdateFleetAttributes gamelift:UpdateFleetCapacity gamelift:UpdateFleetPortSettings

Prefixo do serviço	Ações do
	gamelift:UpdateGameServer gamelift:UpdateGameServerGroup gamelift:UpdateGameSession gamelift:UpdateGameSessionQueue gamelift:UpdateMatchmakingConfiguration gamelift:UpdateRuntimeConfiguration gamelift:UpdateScript gamelift:ValidateMatchmakingRuleSet

Prefixo do serviço	Ações do
geo	geo:AssociateTrackerConsumer
	geo:BatchDeleteDevicePositionHistory
	geo:BatchDeleteGeofence
	geo:BatchEvaluateGeofences
	geo:BatchGetDevicePosition
	geo:BatchPutGeofence
	geo:BatchUpdateDevicePosition
	geo:CalculateRoute
	geo:CalculateRouteMatrix
	geo:CancelJob
	geo:CreateGeofenceCollection
	geo:CreateMap
	geo:CreatePlaceIndex
	geo:CreateRouteCalculator
	geo:CreateTracker
	geo:DeleteGeofenceCollection
	geo:DeleteKey
	geo:DeleteMap
	geo:DeletePlaceIndex
	geo:DeleteRouteCalculator
	geo:DeleteTracker

Prefixo do serviço	Ações do
	geo:DescribeGeofenceCollection
	geo:DescribeKey
	geo:DescribeMap
	geo:DescribePlaceIndex
	geo:DescribeRouteCalculator
	geo:DescribeTracker
	geo:DisassociateTrackerConsumer
	geo:ForecastGeofenceEvents
	geo:GetDevicePosition
	geo:GetDevicePositionHistory
	geo:GetGeofence
	geo:GetJob
	geo:GetMapGlyphs
	geo:GetMapSprites
	geo:GetMapStyleDescriptor
	geo:GetMapTile
	geo:GetPlace
	geo:ListDevicePositions
	geo:ListGeofenceCollections
	geo:ListGeofences
	geo:ListJobs

Prefixo do serviço	Ações do
	geo:ListKeys
	geo:ListMaps
	geo:ListPlaceIndexes
	geo:ListRouteCalculators
	geo:ListTrackerConsumers
	geo:ListTrackers
	geo:PutGeofence
	geo:SearchPlaceIndexForPosition
	geo:SearchPlaceIndexForSuggestions
	geo:SearchPlaceIndexForText
	geo:UpdateGeofenceCollection
	geo:UpdateKey
	geo:UpdateMap
	geo:UpdatePlaceIndex
	geo:UpdateRouteCalculator
	geo:UpdateTracker
	geo:VerifyDevicePosition

Prefixo do serviço	Ações do
glacier	glacier:AbortMultipartUpload glacier:AbortVaultLock glacier:CompleteMultipartUpload glacier:CompleteVaultLock glacier:CreateVault glacier:DeleteArchive glacier:DeleteVault glacier:DeleteVaultAccessPolicy glacier:DeleteVaultNotifications glacier:DescribeJob glacier:DescribeVault glacier:GetDataRetrievalPolicy glacier:GetJobOutput glacier:GetVaultAccessPolicy glacier:GetVaultLock glacier:GetVaultNotifications glacier:InitiateJob glacier:InitiateMultipartUpload glacier:InitiateVaultLock glacier:ListJobs glacier:ListMultipartUploads

Prefixo do serviço	Ações do
	glacier:ListParts glacier:ListProvisionedCapacity glacier:ListVaults glacier:PurchaseProvisionedCapacity glacier:SetDataRetrievalPolicy glacier:SetVaultAccessPolicy glacier:SetVaultNotifications glacier:UploadArchive glacier:UploadMultipartPart

Prefixo do serviço	Ações do
grafana	grafana:AssociateLicense
	grafana:CreateWorkspace
	grafana:CreateWorkspaceApiKey
	grafana:CreateWorkspaceServiceAccount
	grafana:CreateWorkspaceServiceAccountToken
	grafana:DeleteWorkspace
	grafana:DeleteWorkspaceApiKey
	grafana:DeleteWorkspaceServiceAccount
	grafana:DeleteWorkspaceServiceAccountToken
	grafana:DescribeWorkspace
	grafana:DescribeWorkspaceAuthentication
	grafana:DescribeWorkspaceConfiguration
	grafana:DisassociateLicense
	grafana:ListPermissions
	grafana:ListVersions
	grafana:ListWorkspaceServiceAccountTokens
	grafana:ListWorkspaceServiceAccounts
	grafana:ListWorkspaces
	grafana:UpdatePermissions
	grafana:UpdateWorkspace
	grafana:UpdateWorkspaceAuthentication

Prefixo do serviço	Ações do
	grafana:UpdateWorkspaceConfiguration

Prefixo do serviço	Ações do
greengrass	greengrass:AssociateRoleToGroup greengrass:AssociateServiceRoleToAccount greengrass:BatchAssociateClientDeviceWithCoreDevice greengrass:BatchDisassociateClientDeviceFromCoreDevice greengrass:CancelDeployment greengrass:CreateComponentVersion greengrass:CreateConnectorDefinition greengrass:CreateConnectorDefinitionVersion greengrass:CreateCoreDefinition greengrass:CreateCoreDefinitionVersion greengrass:CreateDeployment greengrass:CreateDeviceDefinition greengrass:CreateDeviceDefinitionVersion greengrass:CreateFunctionDefinition greengrass:CreateFunctionDefinitionVersion greengrass:CreateGroup greengrass:CreateGroupCertificateAuthority greengrass:CreateGroupVersion greengrass:CreateLoggerDefinition greengrass:CreateLoggerDefinitionVersion greengrass:CreateResourceDefinition

Prefixo do serviço	Ações do
	greengrass:CreateResourceDefinitionVersion greengrass:CreateSoftwareUpdateJob greengrass:CreateSubscriptionDefinition greengrass:CreateSubscriptionDefinitionVersion greengrass:DeleteComponent greengrass:DeleteConnectorDefinition greengrass:DeleteCoreDefinition greengrass:DeleteCoreDevice greengrass:DeleteDeployment greengrass:DeleteDeviceDefinition greengrass:DeleteFunctionDefinition greengrass:DeleteGroup greengrass:DeleteLoggerDefinition greengrass:DeleteResourceDefinition greengrass:DeleteSubscriptionDefinition greengrass:DescribeComponent greengrass:DisassociateRoleFromGroup greengrass:DisassociateServiceRoleFromAccount greengrass:GetAssociatedRole greengrass:GetBulkDeploymentStatus greengrass:GetComponent

Prefixo do serviço	Ações do
	greengrass:GetComponentVersionArtifact greengrass:GetConnectivityInfo greengrass:GetConnectorDefinition greengrass:GetConnectorDefinitionVersion greengrass:GetCoreDefinition greengrass:GetCoreDefinitionVersion greengrass:GetCoreDevice greengrass:GetDeployment greengrass:GetDeploymentStatus greengrass:GetDeviceDefinition greengrass:GetDeviceDefinitionVersion greengrass:GetFunctionDefinition greengrass:GetFunctionDefinitionVersion greengrass:GetGroup greengrass:GetGroupCertificateAuthority greengrass:GetGroupCertificateConfiguration greengrass:GetGroupVersion greengrass:GetLoggerDefinition greengrass:GetLoggerDefinitionVersion greengrass:GetResourceDefinition greengrass:GetResourceDefinitionVersion

Prefixo do serviço	Ações do
	greengrass:GetServiceRoleForAccount greengrass:GetSubscriptionDefinition greengrass:GetSubscriptionDefinitionVersion greengrass:GetThingRuntimeConfiguration greengrass:ListBulkDeploymentDetailedReports greengrass:ListBulkDeployments greengrass:ListClientDevicesAssociatedWithCoreDevice greengrass:ListComponentVersions greengrass:ListComponents greengrass:ListConnectorDefinitionVersions greengrass:ListConnectorDefinitions greengrass:ListCoreDefinitionVersions greengrass:ListCoreDefinitions greengrass:ListCoreDevices greengrass:ListDeployments greengrass:ListDeviceDefinitionVersions greengrass:ListDeviceDefinitions greengrass:ListEffectiveDeployments greengrass:ListFunctionDefinitionVersions greengrass:ListFunctionDefinitions greengrass:ListGroupCertificateAuthorities

Prefixo do serviço	Ações do
	greengrass:ListGroupVersions greengrass:ListGroups greengrass:ListInstalledComponents greengrass:ListLoggerDefinitionVersions greengrass:ListLoggerDefinitions greengrass:ListResourceDefinitionVersions greengrass:ListResourceDefinitions greengrass:ListSubscriptionDefinitionVersions greengrass:ListSubscriptionDefinitions greengrass:ResetDeployments greengrass:StartBulkDeployment greengrass:StopBulkDeployment greengrass:UpdateConnectivityInfo greengrass:UpdateConnectorDefinition greengrass:UpdateCoreDefinition greengrass:UpdateDeviceDefinition greengrass:UpdateFunctionDefinition greengrass:UpdateGroup greengrass:UpdateGroupCertificateConfiguration greengrass:UpdateLoggerDefinition greengrass:UpdateResourceDefinition

Prefixo do serviço	Ações do
	greengrass:UpdateSubscriptionDefinition greengrass:UpdateThingRuntimeConfiguration

Prefixo do serviço	Ações do
groundstation	groundstation:CancelContact
	groundstation:CreateConfig
	groundstation:CreateDataflowEndpointGroup
	groundstation:CreateDataflowEndpointGroupV2
	groundstation:CreateEphemeris
	groundstation:CreateMissionProfile
	groundstation:DeleteConfig
	groundstation:DeleteDataflowEndpointGroup
	groundstation:DeleteEphemeris
	groundstation:DeleteMissionProfile
	groundstation:DescribeContact
	groundstation:DescribeContactVersion
	groundstation:DescribeEphemeris
	groundstation:GetConfig
	groundstation:GetDataflowEndpointGroup
	groundstation:GetMinuteUsage
	groundstation:GetMissionProfile
	groundstation:GetSatellite
	groundstation:ListAntennas
	groundstation:ListConfigs
	groundstation:ListContacts

Prefixo do serviço	Ações do
	groundstation:ListDataflowEndpointGroups
	groundstation:ListEphemerides
	groundstation:ListGroundStationReservations
	groundstation:ListGroundStations
	groundstation:ListMissionProfiles
	groundstation:ListSatellites
	groundstation:RegisterAgent
	groundstation:ReserveContact
	groundstation:UpdateAgentStatus
	groundstation:UpdateConfig
	groundstation:UpdateEphemeris
	groundstation:UpdateMissionProfile

Prefixo do serviço	Ações do
guardduty	guardduty:AcceptAdministratorInvitation guardduty:AcceptInvitation guardduty:ArchiveFindings guardduty:CreateDetector guardduty:CreateFilter guardduty:CreateIPSet guardduty:CreateInvestigation guardduty:CreateMalwareProtectionPlan guardduty:CreateMembers guardduty:CreatePublishingDestination guardduty:CreateSampleFindings guardduty:CreateThreatEntitySet guardduty:CreateThreatIntelSet guardduty:CreateTrustedEntitySet guardduty:DeclineInvitations guardduty:DeleteDetector guardduty:DeleteFilter guardduty:DeleteIPSet guardduty:DeleteInvitations guardduty:DeleteMalwareProtectionPlan guardduty:DeleteMembers

Prefixo do serviço	Ações do
	guardduty:DeletePublishingDestination guardduty:DeleteThreatEntitySet guardduty:DeleteThreatIntelSet guardduty:DeleteTrustedEntitySet guardduty:DescribeMalwareScans guardduty:DescribeOrganizationConfiguration guardduty:DescribePublishingDestination guardduty:DisableOrganizationAdminAccount guardduty:DisassociateFromAdministratorAccount guardduty:DisassociateFromMasterAccount guardduty:DisassociateMembers guardduty:EnableOrganizationAdminAccount guardduty:GetAdministratorAccount guardduty:GetCoverageStatistics guardduty:GetDetector guardduty:GetFilter guardduty:GetFindings guardduty:GetFindingsStatistics guardduty:GetIPSet guardduty:GetInvestigation guardduty:GetInvitationsCount

Prefixo do serviço	Ações do
	guardduty:GetMalwareProtectionPlan guardduty:GetMalwareScan guardduty:GetMalwareScanSettings guardduty:GetMasterAccount guardduty:GetMemberDetectors guardduty:GetMembers guardduty:GetOrganizationStatistics guardduty:GetRemainingFreeTrialDays guardduty:GetThreatEntitySet guardduty:GetThreatIntelSet guardduty:GetTrustedEntitySet guardduty:GetUsageStatistics guardduty:InviteMembers guardduty:ListCoverage guardduty:ListDetectors guardduty:ListFilters guardduty:ListFindings guardduty:ListIPSets guardduty:ListInvestigations guardduty:ListInvitations guardduty:ListMalwareProtectionPlans

Prefixo do serviço	Ações do
	guardduty:ListMalwareScans guardduty:ListMembers guardduty:ListOrganizationAdminAccounts guardduty:ListPublishingDestinations guardduty:ListThreatEntitySets guardduty:ListThreatIntelSets guardduty:ListTrustedEntitySets guardduty:StartMalwareScan guardduty:StartMonitoringMembers guardduty:StopMonitoringMembers guardduty:UnarchiveFindings guardduty:UpdateDetector guardduty:UpdateFilter guardduty:UpdateFindingsFeedback guardduty:UpdateIPSet guardduty:UpdateMalwareProtectionPlan guardduty:UpdateMalwareScanSettings guardduty:UpdateMemberDetectors guardduty:UpdateOrganizationConfiguration guardduty:UpdatePublishingDestination guardduty:UpdateThreatEntitySet

Prefixo do serviço	Ações do
	guardduty:UpdateThreatIntelSet guardduty:UpdateTrustedEntitySet

Prefixo do serviço	Ações do
healthlake	healthlake:CancelFHIRExportJobWithDelete
	healthlake:CreateDataTransformationProfile
	healthlake:CreateFHIRDatastore
	healthlake:CreateResource
	healthlake:DeleteDataTransformationProfile
	healthlake:DeleteFHIRDatastore
	healthlake:DeleteResource
	healthlake:DescribeDataTransformationJob
	healthlake:DescribeFHIRDatastore
	healthlake:DescribeFHIRExportJob
	healthlake:DescribeFHIRExportJobWithGet
	healthlake:DescribeFHIRImportJob
	healthlake:GetCapabilities
	healthlake:GetDataTransformationProfile
	healthlake:ListDataTransformationJobs
	healthlake:ListDataTransformationProfileVersions
	healthlake:ListDataTransformationProfiles
	healthlake:ListFHIRDatastores
	healthlake:ListFHIRExportJobs
	healthlake:ListFHIRImportJobs
	healthlake:PublishDataTransformationProfile

Prefixo do serviço	Ações do
	healthlake:ReadResource
	healthlake:SearchEverything
	healthlake:SearchWithGet
	healthlake:SearchWithPost
	healthlake:StartFHIRExportJob
	healthlake:StartFHIRExportJobWithPost
	healthlake:StartFHIRImportJob
	healthlake:UpdateDataTransformationProfile
	healthlake:UpdateFHIRDatastore
	healthlake:UpdateProfileWithAgent
	healthlake:UpdateResource

Prefixo do serviço	Ações do
honeycode	honeycode:BatchCreateTableRows honeycode:BatchDeleteTableRows honeycode:BatchUpdateTableRows honeycode:BatchUpsertTableRows honeycode:DescribeTableDataImportJob honeycode:GetScreenData honeycode:InvokeScreenAutomation honeycode:ListTableColumns honeycode:ListTableRows honeycode:ListTables honeycode:QueryTableRows honeycode:StartTableDataImportJob

Prefixo do serviço	Ações do
iam	iam:AddClientIDToOpenIDConnectProvider
	iam:AddRoleToInstanceProfile
	iam:AddUserToGroup
	iam:AttachGroupPolicy
	iam:AttachRolePolicy
	iam:AttachUserPolicy
	iam:ChangePassword
	iam:CreateAccessKey
	iam:CreateAccountAlias
	iam:CreateGroup
	iam:CreateInstanceProfile
	iam:CreateLoginProfile
	iam:CreateOpenIDConnectProvider
	iam:CreatePolicy
	iam:CreatePolicyVersion
	iam:CreateRole
	iam:CreateSAMLProvider
	iam:CreateServiceLinkedRole
	iam:CreateServiceSpecificCredential
	iam:CreateUser
	iam:CreateVirtualMFADevice

Prefixo do serviço	Ações do
	iam:DeactivateMFADevice
	iam:DeleteAccessKey
	iam:DeleteAccountAlias
	iam:DeleteAccountPasswordPolicy
	iam:DeleteCloudFrontPublicKey
	iam:DeleteGroup
	iam:DeleteGroupPolicy
	iam:DeleteInstanceProfile
	iam:DeleteLoginProfile
	iam:DeleteOpenIDConnectProvider
	iam:DeletePolicy
	iam:DeletePolicyVersion
	iam:DeleteRole
	iam:DeleteRolePermissionsBoundary
	iam:DeleteRolePolicy
	iam:DeleteSAMLProvider
	iam:DeleteSSHPublicKey
	iam:DeleteServerCertificate
	iam:DeleteServiceLinkedRole
	iam:DeleteServiceSpecificCredential
	iam:DeleteSigningCertificate

Prefixo do serviço	Ações do
	iam:DeleteUser
	iam:DeleteUserPermissionsBoundary
	iam:DeleteUserPolicy
	iam:DeleteVirtualMFADevice
	iam:DetachGroupPolicy
	iam:DetachRolePolicy
	iam:DetachUserPolicy
	iam:DisableOrganizationsRootCredentialsManagement
	iam:DisableOrganizationsRootSessions
	iam:DisableOutboundWebIdentityFederation
	iam:EnableMFADevice
	iam:EnableOrganizationsRootCredentialsManagement
	iam:EnableOrganizationsRootSessions
	iam:EnableOutboundWebIdentityFederation
	iam:GenerateCredentialReport
	iam:GenerateOrganizationsAccessReport
	iam:GenerateServiceLastAccessedDetails
	iam:GetAccessKeyLastUsed
	iam:GetAccountAuthorizationDetails
	iam:GetAccountEmailAddress
	iam:GetAccountName

Prefixo do serviço	Ações do
	iam:GetAccountPasswordPolicy
	iam:GetAccountSummary
	iam:GetCloudFrontPublicKey
	iam:GetContextKeysForCustomPolicy
	iam:GetContextKeysForPrincipalPolicy
	iam:GetCredentialReport
	iam:GetGroup
	iam:GetGroupPolicy
	iam:GetInstanceProfile
	iam:GetLoginProfile
	iam:GetMFADevice
	iam:GetOpenIDConnectProvider
	iam:GetOrganizationsAccessReport
	iam:GetOutboundWebIdentityFederationInfo
	iam:GetPolicy
	iam:GetPolicyVersion
	iam:GetRole
	iam:GetRolePolicy
	iam:GetSAMLProvider
	iam:GetSSHPublicKey
	iam:GetServerCertificate

Prefixo do serviço	Ações do
	iam:GetServiceLastAccessedDetails
	iam:GetServiceLastAccessedDetailsWithEntities
	iam:GetServiceLinkedRoleDeletionStatus
	iam:GetUser
	iam:GetUserPolicy
	iam:ListAccessKeys
	iam:ListAccountAliases
	iam:ListAttachedGroupPolicies
	iam:ListAttachedRolePolicies
	iam:ListAttachedUserPolicies
	iam:ListCloudFrontPublicKeys
	iam:ListDelegationRequests
	iam:ListEntitiesForPolicy
	iam:ListGroupPolicies
	iam:ListGroups
	iam:ListGroupsForUser
	iam:ListInstanceProfiles
	iam:ListInstanceProfilesForRole
	iam:ListMFADevices
	iam:ListOpenIDConnectProviders
	iam:ListOrganizationsFeatures

Prefixo do serviço	Ações do
	iam:ListPolicies
	iam:ListPoliciesGrantingServiceAccess
	iam:ListPolicyVersions
	iam:ListRolePolicies
	iam:ListRoles
	iam:ListSAMLProviders
	iam:ListSSHPublicKeys
	iam:ListSTSRegionalEndpointsStatus
	iam:ListServerCertificates
	iam:ListServiceSpecificCredentials
	iam:ListSigningCertificates
	iam:ListUserPolicies
	iam:ListUsers
	iam:ListVirtualMFADevices
	iam:PutGroupPolicy
	iam:PutRolePermissionsBoundary
	iam:PutRolePolicy
	iam:PutUserPermissionsBoundary
	iam:PutUserPolicy
	iam:RemoveClientIDFromOpenIDConnectProvider
	iam:RemoveRoleFromInstanceProfile

Prefixo do serviço	Ações do
	iam:RemoveUserFromGroup
	iam:ResetServiceSpecificCredential
	iam:ResyncMFADevice
	iam:SetDefaultPolicyVersion
	iam:SetSTSRegionalEndpointStatus
	iam:SetSecurityTokenServicePreferences
	iam:SimulateCustomPolicy
	iam:SimulatePrincipalPolicy
	iam:UpdateAccessKey
	iam:UpdateAccountEmailAddress
	iam:UpdateAccountName
	iam:UpdateAccountPasswordPolicy
	iam:UpdateAssumeRolePolicy
	iam:UpdateCloudFrontPublicKey
	iam:UpdateGroup
	iam:UpdateLoginProfile
	iam:UpdateOpenIDConnectProviderThumbprint
	iam:UpdateRole
	iam:UpdateRoleDescription
	iam:UpdateSAMLProvider
	iam:UpdateSSHPublicKey

Prefixo do serviço	Ações do
	iam:UpdateServerCertificate
	iam:UpdateServiceSpecificCredential
	iam:UpdateSigningCertificate
	iam:UpdateUser
	iam:UploadCloudFrontPublicKey
	iam:UploadSSHPublicKey
	iam:UploadServerCertificate
	iam:UploadSigningCertificate

Prefixo do serviço	Ações do
identitystore	identitystore:CreateGroup identitystore:CreateGroupMembership identitystore:CreateUser identitystore:DeleteGroup identitystore:DeleteGroupMembership identitystore:DeleteUser identitystore:DescribeGroup identitystore:DescribeGroupMembership identitystore:DescribeUser identitystore:GetGroupId identitystore:GetGroupMembershipId identitystore:GetUserId identitystore:IsMemberInGroups identitystore:ListGroupMemberships identitystore:ListGroupMembershipsForMember identitystore:ListGroups identitystore:ListUsers identitystore:UpdateGroup identitystore:UpdateUser

Prefixo do serviço	Ações do
imagebuilder	imagebuilder:CancelImageCreation
	imagebuilder:CancelLifecycleExecution
	imagebuilder:CreateComponent
	imagebuilder:CreateContainerRecipe
	imagebuilder:CreateDistributionConfiguration
	imagebuilder:CreateImage
	imagebuilder:CreateImagePipeline
	imagebuilder:CreateImageRecipe
	imagebuilder:CreateInfrastructureConfiguration
	imagebuilder:CreateLifecyclePolicy
	imagebuilder:CreateWorkflow
	imagebuilder:DeleteComponent
	imagebuilder:DeleteContainerRecipe
	imagebuilder:DeleteDistributionConfiguration
	imagebuilder:DeleteImage
	imagebuilder:DeleteImagePipeline
	imagebuilder:DeleteImageRecipe
	imagebuilder:DeleteInfrastructureConfiguration
	imagebuilder:DeleteLifecyclePolicy
	imagebuilder:DeleteWorkflow
	imagebuilder:DistributeImage

Prefixo do serviço	Ações do
	imagebuilder:GetComponentPolicy
	imagebuilder:GetContainerRecipePolicy
	imagebuilder:GetImagePolicy
	imagebuilder:GetImageRecipePolicy
	imagebuilder:GetLifecycleExecution
	imagebuilder:GetLifecyclePolicy
	imagebuilder:GetMarketplaceResource
	imagebuilder:GetWorkflowExecution
	imagebuilder:GetWorkflowStepExecution
	imagebuilder:ImportComponent
	imagebuilder:ImportDiskImage
	imagebuilder:ImportVmImage
	imagebuilder:ListComponentBuildVersions
	imagebuilder:ListComponents
	imagebuilder:ListContainerRecipes
	imagebuilder:ListDistributionConfigurations
	imagebuilder:ListImageBuildVersions
	imagebuilder:ListImagePackages
	imagebuilder:ListImagePipelineImages
	imagebuilder:ListImagePipelines
	imagebuilder:ListImageRecipes

Prefixo do serviço	Ações do
	imagebuilder:ListImageScanFindingAggregations
	imagebuilder:ListImageScanFindings
	imagebuilder:ListImages
	imagebuilder:ListInfrastructureConfigurations
	imagebuilder:ListLifecycleExecutionResources
	imagebuilder:ListLifecycleExecutions
	imagebuilder:ListLifecyclePolicies
	imagebuilder:ListWaitingWorkflowSteps
	imagebuilder:ListWorkflowExecutions
	imagebuilder:ListWorkflowStepExecutions
	imagebuilder:ListWorkflows
	imagebuilder:PutComponentPolicy
	imagebuilder:PutContainerRecipePolicy
	imagebuilder:PutImagePolicy
	imagebuilder:PutImageRecipePolicy
	imagebuilder:RetryImage
	imagebuilder:SendWorkflowStepAction
	imagebuilder:StartImagePipelineExecution
	imagebuilder:StartResourceStateUpdate
	imagebuilder:UpdateDistributionConfiguration
	imagebuilder:UpdateImagePipeline

Prefixo do serviço	Ações do
	imagebuilder:UpdateInfrastructureConfiguration

Prefixo do serviço	Ações do
inspector	inspector:AddAttributesToFindings inspector:CreateAssessmentTarget inspector:CreateAssessmentTemplate inspector:CreateExclusionsPreview inspector:CreateResourceGroup inspector:DeleteAssessmentRun inspector:DeleteAssessmentTarget inspector:DeleteAssessmentTemplate inspector:DescribeAssessmentRuns inspector:DescribeAssessmentTargets inspector:DescribeAssessmentTemplates inspector:DescribeCrossAccountAccessRole inspector:DescribeExclusions inspector:DescribeFindings inspector:DescribeResourceGroups inspector:DescribeRulesPackages inspector:GetAssessmentReport inspector:GetExclusionsPreview inspector:GetTelemetryMetadata inspector:ListAssessmentRunAgents inspector:ListAssessmentRuns

Prefixo do serviço	Ações do
	inspector:ListAssessmentTargets inspector:ListAssessmentTemplates inspector:ListEventSubscriptions inspector:ListExclusions inspector:ListFindings inspector:ListRulesPackages inspector:PreviewAgents inspector:RegisterCrossAccountAccessRole inspector:RemoveAttributesFromFindings inspector:StartAssessmentRun inspector:StopAssessmentRun inspector:SubscribeToEvent inspector:UnsubscribeFromEvent inspector:UpdateAssessmentTarget

Prefixo do serviço	Ações do
inspector2	inspector2:AssociateMember
	inspector2:BatchGetAccountStatus
	inspector2:BatchGetCodeSnippet
	inspector2:BatchGetFindingDetails
	inspector2:BatchGetFreeTrialInfo
	inspector2:BatchGetMemberEc2DeepInspectionStatus
	inspector2:BatchUpdateMemberEc2DeepInspectionStatus
	inspector2:CancelFindingsReport
	inspector2:CancelSbomExport
	inspector2:CreateCisScanConfiguration
	inspector2:CreateCodeSecurityIntegration
	inspector2:CreateConnector
	inspector2:CreateFilter
	inspector2:CreateFindingsReport
	inspector2:CreateSbomExport
	inspector2:DeleteCisScanConfiguration
	inspector2:DeleteCodeSecurityIntegration
	inspector2:DeleteConnector
	inspector2:DeleteFilter
	inspector2:DescribeOrganizationConfiguration
	inspector2:Disable

Prefixo do serviço	Ações do
	inspector2:DisableDelegatedAdminAccount
	inspector2:DisassociateMember
	inspector2:Enable
	inspector2:EnableDelegatedAdminAccount
	inspector2:GetCisScanReport
	inspector2:GetCisScanResultDetails
	inspector2:GetClustersForImage
	inspector2:GetCodeSecurityIntegration
	inspector2:GetCodeSecurityScan
	inspector2:GetConfiguration
	inspector2:GetDelegatedAdminAccount
	inspector2:GetEc2DeepInspectionConfiguration
	inspector2:GetEncryptionKey
	inspector2:GetFindingsReportStatus
	inspector2:GetMember
	inspector2:GetSbomExport
	inspector2:ListAccountPermissions
	inspector2:ListCisScanConfigurations
	inspector2:ListCisScanResultsAggregatedByChecks
	inspector2:ListCisScanResultsAggregatedByTargetResource
	inspector2:ListCisScans

Prefixo do serviço	Ações do
	inspector2:ListCodeSecurityIntegrations
	inspector2:ListCodeSecurityScanConfigurations
	inspector2:ListConnectorScanConfigurations
	inspector2:ListConnectors
	inspector2:ListCoverage
	inspector2:ListCoverageStatistics
	inspector2:ListDelegatedAdminAccounts
	inspector2:ListFilters
	inspector2:ListFindingAggregations
	inspector2:ListFindings
	inspector2:ListMembers
	inspector2:ListUsageTotals
	inspector2:ResetEncryptionKey
	inspector2:SearchVulnerabilities
	inspector2:SendCisSessionHealth
	inspector2:SendCisSessionTelemetry
	inspector2:StartCisSession
	inspector2:StartCodeSecurityScan
	inspector2:StopCisSession
	inspector2:UpdateCisScanConfiguration
	inspector2:UpdateCodeSecurityIntegration

Prefixo do serviço	Ações do
	inspector2:UpdateConfiguration
	inspector2:UpdateConnector
	inspector2:UpdateEc2DeepInspectionConfiguration
	inspector2:UpdateEncryptionKey
	inspector2:UpdateFilter
	inspector2:UpdateOrgEc2DeepInspectionConfiguration
	inspector2:UpdateOrganizationConfiguration

Prefixo do serviço	Ações do
iot	iot:AcceptCertificateTransfer iot:AddThingToBillingGroup iot:AddThingToThingGroup iot:AssociateSbomWithPackageVersion iot:AssociateTargetsWithJob iot:AttachPolicy iot:AttachPrincipalPolicy iot:AttachSecurityProfile iot:AttachThingPrincipal iot:CancelAuditMitigationActionsTask iot:CancelAuditTask iot:CancelCertificateTransfer iot:CancelDetectMitigationActionsTask iot:CancelJob iot:CancelJobExecution iot:ClearDefaultAuthorizer iot:ConfirmTopicRuleDestination iot:CreateAuditSuppression iot:CreateAuthorizer iot:CreateBillingGroup iot:CreateCertificateFromCsr

Prefixo do serviço	Ações do
	iot:CreateCertificateProvider iot:CreateCommand iot:CreateCustomMetric iot:CreateDimension iot:CreateDomainConfiguration iot:CreateDynamicThingGroup iot:CreateFleetMetric iot:CreateJob iot:CreateJobTemplate iot:CreateKeysAndCertificate iot:CreateMitigationAction iot:CreateOTAUpdate iot:CreatePackage iot:CreatePackageVersion iot:CreatePolicy iot:CreatePolicyVersion iot:CreateProvisioningClaim iot:CreateProvisioningTemplate iot:CreateProvisioningTemplateVersion iot:CreateRoleAlias iot:CreateScheduledAudit

Prefixo do serviço	Ações do
	iot:CreateSecurityProfile iot:CreateStream iot:CreateThing iot:CreateThingGroup iot:CreateThingType iot:CreateTopicRule iot:CreateTopicRuleDestination iot:DeleteAccountAuditConfiguration iot:DeleteAuditSuppression iot:DeleteAuthorizer iot:DeleteBillingGroup iot:DeleteCACertificate iot:DeleteCertificate iot:DeleteCertificateProvider iot:DeleteCommand iot:DeleteCustomMetric iot:DeleteDimension iot:DeleteDomainConfiguration iot:DeleteDynamicThingGroup iot:DeleteFleetMetric iot:DeleteJob

Prefixo do serviço	Ações do
	iot:DeleteJobExecution iot:DeleteJobTemplate iot:DeleteMitigationAction iot:DeleteOTAUpdate iot:DeletePackage iot:DeletePackageVersion iot:DeletePolicy iot:DeletePolicyVersion iot:DeleteProvisioningTemplate iot:DeleteProvisioningTemplateVersion iot:DeleteRegistrationCode iot:DeleteRoleAlias iot:DeleteScheduledAudit iot:DeleteSecurityProfile iot:DeleteStream iot:DeleteThing iot:DeleteThingGroup iot:DeleteThingType iot:DeleteTopicRule iot:DeleteTopicRuleDestination iot:DeleteV2LoggingLevel

Prefixo do serviço	Ações do
	iot:DeprecateThingType iot:DescribeAccountAuditConfiguration iot:DescribeAuditFinding iot:DescribeAuditMitigationActionsTask iot:DescribeAuditSuppression iot:DescribeAuditTask iot:DescribeAuthorizer iot:DescribeBillingGroup iot:DescribeCACertificate iot:DescribeCertificate iot:DescribeCertificateProvider iot:DescribeCustomMetric iot:DescribeDefaultAuthorizer iot:DescribeDetectMitigationActionsTask iot:DescribeDimension iot:DescribeDomainConfiguration iot:DescribeEncryptionConfiguration iot:DescribeEndpoint iot:DescribeEventConfigurations iot:DescribeFleetMetric iot:DescribeIndex

Prefixo do serviço	Ações do
	iot:DescribeJob iot:DescribeJobExecution iot:DescribeJobTemplate iot:DescribeManagedJobTemplate iot:DescribeMitigationAction iot:DescribeProvisioningTemplate iot:DescribeProvisioningTemplateVersion iot:DescribeRoleAlias iot:DescribeScheduledAudit iot:DescribeSecurityProfile iot:DescribeStream iot:DescribeThing iot:DescribeThingGroup iot:DescribeThingRegistrationTask iot:DescribeThingType iot:DetachPolicy iot:DetachPrincipalPolicy iot:DetachSecurityProfile iot:DetachThingPrincipal iot:DisableTopicRule iot:DisassociateSbomFromPackageVersion

Prefixo do serviço	Ações do
	iot:EnableTopicRule iot:GetBehaviorModelTrainingSummaries iot:GetBucketsAggregation iot:GetCardinality iot:GetCommand iot:GetEffectivePolicies iot:GetJobDocument iot:GetLoggingOptions iot:GetOTAUpdate iot:GetPackage iot:GetPackageConfiguration iot:GetPackageVersion iot:GetPercentiles iot:GetPolicy iot:GetPolicyVersion iot:GetRegistrationCode iot:GetStatistics iot:GetThingConnectivityData iot:GetTopicRule iot:GetTopicRuleDestination iot:GetV2LoggingOptions

Prefixo do serviço	Ações do
	iot:ListActiveViolations iot:ListAttachedPolicies iot:ListAuditFindings iot:ListAuditMitigationActionsExecutions iot:ListAuditMitigationActionsTasks iot:ListAuditSuppressions iot:ListAuditTasks iot:ListAuthorizers iot:ListBillingGroups iot:ListCACertificates iot:ListCertificateProviders iot:ListCertificates iot:ListCertificatesByCA iot:ListCommands iot:ListCustomMetrics iot:ListDetectMitigationActionsExecutions iot:ListDetectMitigationActionsTasks iot:ListDimensions iot:ListDomainConfigurations iot:ListFleetMetrics iot:ListIndices

Prefixo do serviço	Ações do
	iot:ListJobExecutionsForJob iot:ListJobExecutionsForThing iot:ListJobTemplates iot:ListJobs iot:ListManagedJobTemplates iot:ListMetricValues iot:ListMitigationActions iot:ListOTAUpdates iot:ListOutgoingCertificates iot:ListPackageVersions iot:ListPackages iot:ListPolicies iot:ListPolicyPrincipals iot:ListPolicyVersions iot:ListPrincipalPolicies iot:ListPrincipalThings iot:ListPrincipalThingsV2 iot:ListProvisioningTemplateVersions iot:ListProvisioningTemplates iot:ListRelatedResourcesForAuditFinding iot:ListRoleAliases

Prefixo do serviço	Ações do
	iot:ListSbomValidationResults iot:ListScheduledAudits iot:ListSecurityProfiles iot:ListSecurityProfilesForTarget iot:ListStreams iot:ListTargetsForPolicy iot:ListTargetsForSecurityProfile iot:ListThingGroups iot:ListThingGroupsForThing iot:ListThingPrincipals iot:ListThingPrincipalsV2 iot:ListThingRegistrationTaskReports iot:ListThingRegistrationTasks iot:ListThingTypes iot:ListThings iot:ListThingsInBillingGroup iot:ListThingsInThingGroup iot:ListTopicRuleDestinations iot:ListTopicRules iot:ListV2LoggingLevels iot:ListViolationEvents

Prefixo do serviço	Ações do
	iot:PutVerificationStateOnViolation iot:RegisterCACertificate iot:RegisterCertificate iot:RegisterCertificateWithoutCA iot:RegisterThing iot:RejectCertificateTransfer iot:RemoveThingFromBillingGroup iot:RemoveThingFromThingGroup iot:ReplaceTopicRule iot:SearchIndex iot:SetDefaultAuthorizer iot:SetDefaultPolicyVersion iot:SetLoggingOptions iot:SetV2LoggingLevel iot:SetV2LoggingOptions iot:StartAuditMitigationActionsTask iot:StartDetectMitigationActionsTask iot:StartOnDemandAuditTask iot:StartThingRegistrationTask iot:StopThingRegistrationTask iot:TestAuthorization

Prefixo do serviço	Ações do
	iot:TestInvokeAuthorizer iot:TransferCertificate iot:UpdateAccountAuditConfiguration iot:UpdateAuditSuppression iot:UpdateAuthorizer iot:UpdateBillingGroup iot:UpdateCACertificate iot:UpdateCertificate iot:UpdateCertificateProvider iot:UpdateCommand iot:UpdateCustomMetric iot:UpdateDimension iot:UpdateDomainConfiguration iot:UpdateDynamicThingGroup iot:UpdateEncryptionConfiguration iot:UpdateEventConfigurations iot:UpdateFleetMetric iot:UpdateIndexingConfiguration iot:UpdateJob iot:UpdateMitigationAction iot:UpdatePackage

Prefixo do serviço	Ações do
	iot:UpdatePackageConfiguration iot:UpdatePackageVersion iot:UpdateProvisioningTemplate iot:UpdateRoleAlias iot:UpdateScheduledAudit iot:UpdateSecurityProfile iot:UpdateStream iot:UpdateThing iot:UpdateThingGroup iot:UpdateThingGroupsForThing iot:UpdateThingType iot:UpdateTopicRuleDestination iot:ValidateSecurityProfileBehaviors

Prefixo do serviço	Ações do
iotanalytics	iotanalytics:CancelPipelineReprocessing
	iotanalytics:CreateChannel
	iotanalytics:CreateDataset
	iotanalytics:CreateDatasetContent
	iotanalytics:CreateDatastore
	iotanalytics:CreatePipeline
	iotanalytics:DeleteChannel
	iotanalytics:DeleteDataset
	iotanalytics:DeleteDatasetContent
	iotanalytics:DeleteDatastore
	iotanalytics:DeletePipeline
	iotanalytics:DescribeChannel
	iotanalytics:DescribeDataset
	iotanalytics:DescribeDatastore
	iotanalytics:DescribeLoggingOptions
	iotanalytics:DescribePipeline
	iotanalytics:GetDatasetContent
	iotanalytics:ListChannels
	iotanalytics:ListDatasetContents
	iotanalytics:ListDatasets
	iotanalytics:ListDatastores

Prefixo do serviço	Ações do
	iotanalytics:ListPipelines iotanalytics:PutLoggingOptions iotanalytics:RunPipelineActivity iotanalytics:SampleChannelData iotanalytics:StartPipelineReprocessing iotanalytics:UpdateChannel iotanalytics:UpdateDataset iotanalytics:UpdateDatastore iotanalytics:UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor:CreateSuiteDefinition iotdeviceadvisor:DeleteSuiteDefinition iotdeviceadvisor:GetEndpoint iotdeviceadvisor:GetSuiteDefinition iotdeviceadvisor:GetSuiteRun iotdeviceadvisor:GetSuiteRunReport iotdeviceadvisor:ListSuiteDefinitions iotdeviceadvisor:ListSuiteRuns iotdeviceadvisor:StartSuiteRun iotdeviceadvisor:StopSuiteRun iotdeviceadvisor:UpdateSuiteDefinition

Prefixo do serviço	Ações do
iotevents	iotevents:BatchAcknowledgeAlarm
	iotevents:BatchDeleteDetector
	iotevents:BatchDisableAlarm
	iotevents:BatchEnableAlarm
	iotevents:BatchResetAlarm
	iotevents:BatchSnoozeAlarm
	iotevents:BatchUpdateDetector
	iotevents:CreateAlarmModel
	iotevents:CreateDetectorModel
	iotevents:CreateInput
	iotevents:DeleteAlarmModel
	iotevents:DeleteDetectorModel
	iotevents:DeleteInput
	iotevents:DescribeAlarm
	iotevents:DescribeAlarmModel
	iotevents:DescribeDetector
	iotevents:DescribeDetectorModel
	iotevents:DescribeDetectorModelAnalysis
	iotevents:DescribeInput
	iotevents:DescribeLoggingOptions
	iotevents:GetDetectorModelAnalysisResults

Prefixo do serviço	Ações do
	iotevents:ListAlarmModelVersions iotevents:ListAlarmModels iotevents:ListAlarms iotevents:ListDetectorModelVersions iotevents:ListDetectorModels iotevents:ListDetectors iotevents:ListInputRoutings iotevents:ListInputs iotevents:PutLoggingOptions iotevents:StartDetectorModelAnalysis iotevents:UpdateAlarmModel iotevents:UpdateDetectorModel iotevents:UpdateInput
iotfleethub	iotfleethub:CreateApplication iotfleethub:DeleteApplication iotfleethub:DescribeApplication iotfleethub:ListApplications iotfleethub:UpdateApplication

Prefixo do serviço	Ações do
iotsitewise	iotsitewise:AssociateAssets
	iotsitewise:AssociateTimeSeriesToAssetProperty
	iotsitewise:BatchAssociateProjectAssets
	iotsitewise:BatchDisassociateProjectAssets
	iotsitewise:CreateAccessPolicy
	iotsitewise:CreateAsset
	iotsitewise:CreateAssetModel
	iotsitewise:CreateAssetModelCompositeModel
	iotsitewise:CreateBulkImportJob
	iotsitewise:CreateComputationModel
	iotsitewise:CreateDashboard
	iotsitewise:CreateDataset
	iotsitewise:CreateGateway
	iotsitewise:CreatePortal
	iotsitewise:CreateProject
	iotsitewise:DeleteAccessPolicy
	iotsitewise:DeleteAsset
	iotsitewise:DeleteAssetModel
	iotsitewise:DeleteAssetModelCompositeModel
	iotsitewise:DeleteComputationModel
	iotsitewise:DeleteDashboard

Prefixo do serviço	Ações do
	iotsitewise:DeleteDataset
	iotsitewise:DeleteGateway
	iotsitewise:DeletePortal
	iotsitewise:DeleteProject
	iotsitewise:DeleteTimeSeries
	iotsitewise:DescribeAccessPolicy
	iotsitewise:DescribeAsset
	iotsitewise:DescribeAssetCompositeModel
	iotsitewise:DescribeAssetModel
	iotsitewise:DescribeAssetModelCompositeModel
	iotsitewise:DescribeAssetModelInterfaceRelationship
	iotsitewise:DescribeAssetProperty
	iotsitewise:DescribeBulkImportJob
	iotsitewise:DescribeComputationModel
	iotsitewise:DescribeComputationModelExecutionSummary
	iotsitewise:DescribeDashboard
	iotsitewise:DescribeDataset
	iotsitewise:DescribeDefaultEncryptionConfiguration
	iotsitewise:DescribeExecution
	iotsitewise:DescribeGateway
	iotsitewise:DescribeGatewayCapabilityConfiguration

Prefixo do serviço	Ações do
	iotsitewise:DescribeLoggingOptions
	iotsitewise:DescribePortal
	iotsitewise:DescribeProject
	iotsitewise:DescribeStorageConfiguration
	iotsitewise:DescribeTimeSeries
	iotsitewise:DisassociateAssets
	iotsitewise:DisassociateTimeSeriesFromAssetProperty
	iotsitewise:ExecuteAction
	iotsitewise:ExecuteQuery
	iotsitewise:ListAccessPolicies
	iotsitewise:ListActions
	iotsitewise:ListAssetModelCompositeModels
	iotsitewise:ListAssetModelProperties
	iotsitewise:ListAssetModels
	iotsitewise:ListAssetProperties
	iotsitewise:ListAssetRelationships
	iotsitewise:ListAssets
	iotsitewise:ListAssociatedAssets
	iotsitewise:ListBulkImportJobs
	iotsitewise:ListCompositionRelationships
	iotsitewise:ListComputationModelDataBindingUsages

Prefixo do serviço	Ações do
	iotsitewise:ListComputationModelResolveToResources
	iotsitewise:ListComputationModels
	iotsitewise:ListDashboards
	iotsitewise:ListDatasets
	iotsitewise:ListExecutions
	iotsitewise:ListGateways
	iotsitewise:ListInterfaceRelationships
	iotsitewise:ListPortals
	iotsitewise:ListProjectAssets
	iotsitewise:ListProjects
	iotsitewise:ListTimeSeries
	iotsitewise:PutDefaultEncryptionConfiguration
	iotsitewise:PutLoggingOptions
	iotsitewise:PutStorageConfiguration
	iotsitewise:UpdateAccessPolicy
	iotsitewise:UpdateAsset
	iotsitewise:UpdateAssetModel
	iotsitewise:UpdateAssetModelCompositeModel
	iotsitewise:UpdateAssetProperty
	iotsitewise:UpdateComputationModel
	iotsitewise:UpdateDashboard

Prefixo do serviço	Ações do
	iotsitewise:UpdateDataset
	iotsitewise:UpdateGateway
	iotsitewise:UpdateGatewayCapabilityConfiguration
	iotsitewise:UpdatePortal
	iotsitewise:UpdateProject

Prefixo do serviço	Ações do
iottwinmaker	iottwinmaker:CancelMetadataTransferJob iottwinmaker:CreateComponentType iottwinmaker:CreateEntity iottwinmaker:CreateMetadataTransferJob iottwinmaker:CreateScene iottwinmaker:CreateSyncJob iottwinmaker:CreateWorkspace iottwinmaker:DeleteComponentType iottwinmaker:DeleteEntity iottwinmaker:DeleteScene iottwinmaker:DeleteSyncJob iottwinmaker:DeleteWorkspace iottwinmaker:ExecuteQuery iottwinmaker:GetMetadataTransferJob iottwinmaker:GetPricingPlan iottwinmaker:GetScene iottwinmaker:GetSyncJob iottwinmaker:ListComponentTypes iottwinmaker:ListComponents iottwinmaker:ListEntities iottwinmaker:ListMetadataTransferJobs

Prefixo do serviço	Ações do
	iottwinmaker:ListProperties iottwinmaker:ListScenes iottwinmaker:ListSyncJobs iottwinmaker:ListSyncResources iottwinmaker:ListWorkspaces iottwinmaker:UpdateComponentType iottwinmaker:UpdateEntity iottwinmaker:UpdatePricingPlan iottwinmaker:UpdateScene iottwinmaker:UpdateWorkspace

Prefixo do serviço	Ações do
iotwireless	iotwireless:AssociateAwsAccountWithPartnerAccount
	iotwireless:AssociateMulticastGroupWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithFuotaTask
	iotwireless:AssociateWirelessDeviceWithMulticastGroup
	iotwireless:AssociateWirelessDeviceWithThing
	iotwireless:AssociateWirelessGatewayWithCertificate
	iotwireless:AssociateWirelessGatewayWithThing
	iotwireless:CancelMulticastGroupSession
	iotwireless:CreateDestination
	iotwireless:CreateDeviceProfile
	iotwireless:CreateFuotaTask
	iotwireless:CreateMulticastGroup
	iotwireless:CreateNetworkAnalyzerConfiguration
	iotwireless:CreateServiceProfile
	iotwireless:CreateWirelessDevice
	iotwireless:CreateWirelessGateway
	iotwireless:CreateWirelessGatewayTask
	iotwireless:CreateWirelessGatewayTaskDefinition
	iotwireless:DeleteDestination
	iotwireless:DeleteDeviceProfile
	iotwireless:DeleteFuotaTask

Prefixo do serviço	Ações do
	iotwireless:DeleteMulticastGroup
	iotwireless:DeleteNetworkAnalyzerConfiguration
	iotwireless:DeleteQueuedMessages
	iotwireless:DeleteServiceProfile
	iotwireless:DeleteWirelessDevice
	iotwireless:DeleteWirelessDeviceImportTask
	iotwireless:DeleteWirelessGateway
	iotwireless:DeleteWirelessGatewayTask
	iotwireless:DeleteWirelessGatewayTaskDefinition
	iotwireless:DeregisterWirelessDevice
	iotwireless:DisassociateAwsAccountFromPartnerAccount
	iotwireless:DisassociateMulticastGroupFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromFuotaTask
	iotwireless:DisassociateWirelessDeviceFromMulticastGroup
	iotwireless:DisassociateWirelessDeviceFromThing
	iotwireless:DisassociateWirelessGatewayFromCertificate
	iotwireless:DisassociateWirelessGatewayFromThing
	iotwireless:GetDestination
	iotwireless:GetDeviceProfile
	iotwireless:GetEventConfigurationByResourceTypes
	iotwireless:GetFuotaTask

Prefixo do serviço	Ações do
	iotwireless:GetLogLevelsByResourceTypes
	iotwireless:GetMetricConfiguration
	iotwireless:GetMetrics
	iotwireless:GetMulticastGroup
	iotwireless:GetMulticastGroupSession
	iotwireless:GetNetworkAnalyzerConfiguration
	iotwireless:GetPartnerAccount
	iotwireless:GetPosition
	iotwireless:GetPositionConfiguration
	iotwireless:GetPositionEstimate
	iotwireless:GetResourceEventConfiguration
	iotwireless:GetResourceLogLevel
	iotwireless:GetResourcePosition
	iotwireless:GetServiceEndpoint
	iotwireless:GetServiceProfile
	iotwireless:GetWirelessDevice
	iotwireless:GetWirelessDeviceImportTask
	iotwireless:GetWirelessDeviceStatistics
	iotwireless:GetWirelessGateway
	iotwireless:GetWirelessGatewayCertificate
	iotwireless:GetWirelessGatewayFirmwareInformation

Prefixo do serviço	Ações do
	iotwireless:GetWirelessGatewayStatistics
	iotwireless:GetWirelessGatewayTask
	iotwireless:GetWirelessGatewayTaskDefinition
	iotwireless:ListDestinations
	iotwireless:ListDeviceProfiles
	iotwireless:ListDevicesForWirelessDeviceImportTask
	iotwireless:ListEventConfigurations
	iotwireless:ListFuotaTasks
	iotwireless:ListMulticastGroups
	iotwireless:ListMulticastGroupsByFuotaTask
	iotwireless:ListNetworkAnalyzerConfigurations
	iotwireless:ListPartnerAccounts
	iotwireless:ListPositionConfigurations
	iotwireless:ListQueuedMessages
	iotwireless:ListServiceProfiles
	iotwireless:ListWirelessDeviceImportTasks
	iotwireless:ListWirelessDevices
	iotwireless:ListWirelessGatewayTaskDefinitions
	iotwireless:ListWirelessGateways
	iotwireless:PutPositionConfiguration
	iotwireless:PutResourceLogLevel

Prefixo do serviço	Ações do
	iotwireless:ResetAllResourceLogLevels
	iotwireless:ResetResourceLogLevel
	iotwireless:SendDataToMulticastGroup
	iotwireless:SendDataToWirelessDevice
	iotwireless:StartBulkAssociateWirelessDeviceWithMulticastGroup
	iotwireless:StartBulkDisassociateWirelessDeviceFromMulticastGroup
	iotwireless:StartFuotaTask
	iotwireless:StartMulticastGroupSession
	iotwireless:StartNetworkAnalyzerStream
	iotwireless:StartSingleWirelessDeviceImportTask
	iotwireless:StartWirelessDeviceImportTask
	iotwireless:TestWirelessDevice
	iotwireless:UpdateDestination
	iotwireless:UpdateEventConfigurationByResourceTypes
	iotwireless:UpdateFuotaTask
	iotwireless:UpdateLogLevelsByResourceTypes
	iotwireless:UpdateMetricConfiguration
	iotwireless:UpdateMulticastGroup
	iotwireless:UpdateNetworkAnalyzerConfiguration
	iotwireless:UpdatePartnerAccount

Prefixo do serviço	Ações do
	iotwireless:UpdatePosition iotwireless:UpdateResourceEventConfiguration iotwireless:UpdateResourcePosition iotwireless:UpdateWirelessDevice iotwireless:UpdateWirelessDeviceImportTask iotwireless:UpdateWirelessGateway

Prefixo do serviço	Ações do
ivs	ivs:BatchGetChannel ivs:BatchGetStreamKey ivs:BatchStartViewerSessionRevocation ivs:CreateAdConfiguration ivs:CreateChannel ivs:CreateEncoderConfiguration ivs:CreateIngestConfiguration ivs:CreateParticipantToken ivs:CreatePlaybackRestrictionPolicy ivs:CreateRecordingConfiguration ivs:CreateStorageConfiguration ivs:CreateStreamKey ivs:DeleteAdConfiguration ivs:DeleteChannel ivs:DeleteEncoderConfiguration ivs:DeleteIngestConfiguration ivs:DeletePlaybackKeyPair ivs:DeletePlaybackRestrictionPolicy ivs:DeletePublicKey ivs:DeleteRecordingConfiguration ivs:DeleteStorageConfiguration

Prefixo do serviço	Ações do
	ivs:DeleteStreamKey ivs:DisconnectParticipant ivs:GetAdConfiguration ivs:GetChannel ivs:GetComposition ivs:GetEncoderConfiguration ivs:GetIngestConfiguration ivs:GetParticipant ivs:GetPlaybackKeyPair ivs:GetPlaybackRestrictionPolicy ivs:GetPublicKey ivs:GetRecordingConfiguration ivs:GetStorageConfiguration ivs:GetStream ivs:GetStreamKey ivs:GetStreamSession ivs:ImportPlaybackKeyPair ivs:ImportPublicKey ivs:InsertAdBreak ivs:ListAdConfigurations ivs:ListChannels

Prefixo do serviço	Ações do
	ivs:ListCompositions ivs:ListEncoderConfigurations ivs:ListIngestConfigurations ivs:ListParticipantEvents ivs:ListParticipantReplicas ivs:ListParticipants ivs:ListPlaybackKeyPairs ivs:ListPlaybackRestrictionPolicies ivs:ListPublicKeys ivs:ListRecordingConfigurations ivs:ListStorageConfigurations ivs:ListStreamKeys ivs:ListStreamSessions ivs:ListStreams ivs:PutMetadata ivs:StartComposition ivs:StartViewerSessionRevocation ivs:StopComposition ivs:StopStream ivs:UpdateAdConfiguration ivs:UpdateChannel

Prefixo do serviço	Ações do
	ivs:UpdateIngestConfiguration ivs:UpdatePlaybackRestrictionPolicy
ivschat	ivschat:CreateChatToken ivschat:CreateLoggingConfiguration ivschat:CreateRoom ivschat:DeleteLoggingConfiguration ivschat:DeleteMessage ivschat:DeleteRoom ivschat:DisconnectUser ivschat:GetLoggingConfiguration ivschat:GetRoom ivschat:ListLoggingConfigurations ivschat:ListRooms ivschat:SendEvent ivschat:UpdateLoggingConfiguration ivschat:UpdateRoom

Prefixo do serviço	Ações do
kafka	kafka:BatchAssociateScramSecret kafka:BatchDisassociateScramSecret kafka:CreateCluster kafka:CreateClusterV2 kafka:CreateConfiguration kafka:CreateReplicator kafka:CreateVpcConnection kafka:DeleteCluster kafka:DeleteClusterPolicy kafka:DeleteConfiguration kafka:DeleteReplicator kafka:DeleteVpcConnection kafka:DescribeCluster kafka:DescribeClusterOperation kafka:DescribeClusterOperationV2 kafka:DescribeClusterV2 kafka:DescribeConfiguration kafka:DescribeConfigurationRevision kafka:DescribeVpcConnection kafka:GetBootstrapBrokers kafka:GetClusterPolicy

Prefixo do serviço	Ações do
	kafka:GetCompatibleKafkaVersions kafka:ListClientVpcConnections kafka:ListClusterOperations kafka:ListClusterOperationsV2 kafka:ListClusters kafka:ListClustersV2 kafka:ListConfigurationRevisions kafka:ListConfigurations kafka:ListKafkaVersions kafka:ListNodes kafka:ListReplicators kafka:ListScramSecrets kafka:ListVpcConnections kafka:PutClusterPolicy kafka:RebootBroker kafka:RejectClientVpcConnection kafka:UpdateBrokerCount kafka:UpdateBrokerStorage kafka:UpdateBrokerType kafka:UpdateClusterConfiguration kafka:UpdateClusterKafkaVersion

Prefixo do serviço	Ações do
	kafka:UpdateConfiguration kafka:UpdateConnectivity kafka:UpdateMonitoring kafka:UpdateRebalancing kafka:UpdateReplicationInfo kafka:UpdateSecurity kafka:UpdateStorage
kafkaconnect	kafkaconnect:CreateConnector kafkaconnect:CreateCustomPlugin kafkaconnect:CreateWorkerConfiguration kafkaconnect:DeleteConnector kafkaconnect:DeleteCustomPlugin kafkaconnect:DeleteWorkerConfiguration kafkaconnect:DescribeConnector kafkaconnect:DescribeCustomPlugin kafkaconnect:DescribeWorkerConfiguration kafkaconnect:ListConnectorOperations kafkaconnect:ListConnectors kafkaconnect:ListCustomPlugins kafkaconnect:ListWorkerConfigurations kafkaconnect:UpdateConnector

Prefixo do serviço	Ações do
kendra	kendra:AssociateEntitiesToExperience kendra:AssociatePersonasToEntities kendra:BatchDeleteDocument kendra:BatchDeleteFeaturedResultsSet kendra:BatchGetDocumentStatus kendra:BatchPutDocument kendra:ClearQuerySuggestions kendra:CreateAccessControlConfiguration kendra:CreateDataSource kendra:CreateExperience kendra:CreateFaq kendra:CreateFeaturedResultsSet kendra:CreateIndex kendra:CreateQuerySuggestionsBlockList kendra:CreateThesaurus kendra:DeleteDataSource kendra:DeleteExperience kendra:DeleteFaq kendra:DeleteIndex kendra:DeletePrincipalMapping kendra:DeleteQuerySuggestionsBlockList

Prefixo do serviço	Ações do
	kendra:DeleteThesaurus kendra:DescribeAccessControlConfiguration kendra:DescribeDataSource kendra:DescribeExperience kendra:DescribeFaq kendra:DescribeFeaturedResultsSet kendra:DescribeIndex kendra:DescribePrincipalMapping kendra:DescribeQuerySuggestionsBlockList kendra:DescribeQuerySuggestionsConfig kendra:DescribeThesaurus kendra:DisassociateEntitiesFromExperience kendra:DisassociatePersonasFromEntities kendra:GetQuerySuggestions kendra:GetSnapshots kendra:ListAccessControlConfigurations kendra:ListDataSourceSyncJobs kendra:ListDataSources kendra:ListEntityPersonas kendra:ListExperienceEntities kendra:ListExperiences

Prefixo do serviço	Ações do
	kendra:ListFaqs kendra:ListFeaturedResultsSets kendra:ListGroupsOlderThanOrderingId kendra:ListIndices kendra:ListQuerySuggestionsBlockLists kendra:ListThesauri kendra:PutPrincipalMapping kendra:Query kendra:Retrieve kendra:StartDataSourceSyncJob kendra:StopDataSourceSyncJob kendra:SubmitFeedback kendra:UpdateDataSource kendra:UpdateExperience kendra:UpdateFeaturedResultsSet kendra:UpdateIndex kendra:UpdateQuerySuggestionsBlockList kendra:UpdateQuerySuggestionsConfig kendra:UpdateThesaurus

Prefixo do serviço	Ações do
kinesis	kinesis:CreateStream
	kinesis:DecreaseStreamRetentionPeriod
	kinesis:DeleteStream
	kinesis:DeregisterStreamConsumer
	kinesis:DescribeAccountSettings
	kinesis:DescribeLimits
	kinesis:DescribeStream
	kinesis:DescribeStreamConsumer
	kinesis:DescribeStreamSummary
	kinesis:DisableEnhancedMonitoring
	kinesis:EnableEnhancedMonitoring
	kinesis:IncreaseStreamRetentionPeriod
	kinesis:ListShards
	kinesis:ListStreamConsumers
	kinesis:ListStreams
	kinesis:MergeShards
	kinesis:RegisterStreamConsumer
	kinesis:SplitShard
	kinesis:StartStreamEncryption
	kinesis:StopStreamEncryption
	kinesis:UpdateAccountSettings

Prefixo do serviço	Ações do
	kinesis:UpdateShardCount kinesis:UpdateStreamMode

Prefixo do serviço	Ações do
kinesisanalytics	kinesisanalytics:AddApplicationCloudWatchLoggingOption kinesisanalytics:AddApplicationInput kinesisanalytics:AddApplicationInputProcessingConfiguration kinesisanalytics:AddApplicationOutput kinesisanalytics:AddApplicationReferenceDataSource kinesisanalytics:AddApplicationVpcConfiguration kinesisanalytics:CreateApplication kinesisanalytics:CreateApplicationPresignedUrl kinesisanalytics:CreateApplicationSnapshot kinesisanalytics:DeleteApplication kinesisanalytics:DeleteApplicationCloudWatchLoggingOption kinesisanalytics:DeleteApplicationInputProcessingConfiguration kinesisanalytics:DeleteApplicationOutput kinesisanalytics:DeleteApplicationReferenceDataSource kinesisanalytics:DeleteApplicationSnapshot kinesisanalytics:DeleteApplicationVpcConfiguration kinesisanalytics:DescribeApplication kinesisanalytics:DescribeApplicationOperation kinesisanalytics:DescribeApplicationSnapshot kinesisanalytics:DescribeApplicationVersion kinesisanalytics:DiscoverInputSchema

Prefixo do serviço	Ações do
	kinesisanalytics:ListApplicationOperations kinesisanalytics:ListApplicationSnapshots kinesisanalytics:ListApplicationVersions kinesisanalytics:ListApplications kinesisanalytics:RollbackApplication kinesisanalytics:StartApplication kinesisanalytics:StopApplication kinesisanalytics:UpdateApplication kinesisanalytics:UpdateApplicationMaintenanceConfiguration

Prefixo do serviço	Ações do
kms	kms:CancelKeyDeletion
	kms:ConnectCustomKeyStore
	kms:CreateAlias
	kms:CreateCustomKeyStore
	kms:CreateGrant
	kms:CreateKey
	kms:Decrypt
	kms:DeleteAlias
	kms:DeleteCustomKeyStore
	kms:DeleteImportedKeyMaterial
	kms:DeriveSharedSecret
	kms:DescribeCustomKeyStores
	kms:DescribeKey
	kms:DisableKey
	kms:DisableKeyRotation
	kms:DisconnectCustomKeyStore
	kms:EnableKey
	kms:EnableKeyRotation
	kms:Encrypt
	kms:GenerateDataKey
	kms:GenerateDataKeyPair

Prefixo do serviço	Ações do
	kms:GenerateDataKeyPairWithoutPlaintext
	kms:GenerateDataKeyWithoutPlaintext
	kms:GenerateMac
	kms:GenerateRandom
	kms:GetKeyLastUsage
	kms:GetKeyPolicy
	kms:GetKeyRotationStatus
	kms:GetParametersForImport
	kms:GetPublicKey
	kms:ImportKeyMaterial
	kms:ListAliases
	kms:ListGrants
	kms:ListKeyPolicies
	kms:ListKeyRotations
	kms:ListKeys
	kms:ListRetirableGrants
	kms:ReplicateKey
	kms:RetireGrant
	kms:RevokeGrant
	kms:RotateKeyOnDemand
	kms:ScheduleKeyDeletion

Prefixo do serviço	Ações do
	kms:Sign
	kms:UpdateAlias
	kms:UpdateCustomKeyStore
	kms:UpdateKeyDescription
	kms:UpdatePrimaryRegion
	kms:Verify
	kms:VerifyMac

Prefixo do serviço	Ações do
lambda	lambda:AddLayerVersionPermission lambda:AddPermission lambda:CreateAlias lambda:CreateCodeSigningConfig lambda:CreateEventSourceMapping lambda:CreateFunction lambda:CreateFunctionUrlConfig lambda:DeleteAlias lambda:DeleteCapacityProvider lambda:DeleteCodeSigningConfig lambda:DeleteEventSourceMapping lambda:DeleteFunction lambda:DeleteFunctionCodeSigningConfig lambda:DeleteFunctionConcurrency lambda:DeleteFunctionEventInvokeConfig lambda:DeleteFunctionUrlConfig lambda:DeleteLayerVersion lambda:DeleteProvisionedConcurrencyConfig lambda:GetAccountSettings lambda:GetAlias lambda:GetCapacityProvider

Prefixo do serviço	Ações do
	lambda:GetCodeSigningConfig lambda:GetEventSourceMapping lambda:GetFunction lambda:GetFunctionCodeSigningConfig lambda:GetFunctionConcurrency lambda:GetFunctionConfiguration lambda:GetFunctionEventInvokeConfig lambda:GetFunctionRecursionConfig lambda:GetFunctionScalingConfig lambda:GetFunctionUrlConfig lambda:GetLayerVersion lambda:GetLayerVersionPolicy lambda:GetPolicy lambda:GetProvisionedConcurrencyConfig lambda:GetRuntimeManagementConfig lambda:ListAliases lambda:ListCapacityProviders lambda:ListCodeSigningConfigs lambda:ListDurableExecutionsByFunction lambda:ListEventSourceMappings lambda:ListFunctionEventInvokeConfigs

Prefixo do serviço	Ações do
	lambda:ListFunctionUrlConfigs lambda:ListFunctions lambda:ListFunctionsByCodeSigningConfig lambda:ListLayerVersions lambda:ListLayers lambda:ListProvisionedConcurrencyConfigs lambda:ListVersionsByFunction lambda:PublishLayerVersion lambda:PublishVersion lambda:PutFunctionCodeSigningConfig lambda:PutFunctionConcurrency lambda:PutFunctionEventInvokeConfig lambda:PutFunctionRecursionConfig lambda:PutFunctionScalingConfig lambda:PutProvisionedConcurrencyConfig lambda:PutRuntimeManagementConfig lambda:RemoveLayerVersionPermission lambda:RemovePermission lambda:UpdateAlias lambda:UpdateCapacityProvider lambda:UpdateCodeSigningConfig

Prefixo do serviço	Ações do
	lambda:UpdateEventSourceMapping lambda:UpdateFunctionCode lambda:UpdateFunctionConfiguration lambda:UpdateFunctionEventInvokeConfig lambda:UpdateFunctionUrlConfig

Prefixo do serviço	Ações do
lex	lex:BatchCreateCustomVocabularyItem
	lex:BatchDeleteCustomVocabularyItem
	lex:BatchUpdateCustomVocabularyItem
	lex:BuildBotLocale
	lex:CreateBotAlias
	lex:CreateBotReplica
	lex:CreateBotVersion
	lex:CreateExport
	lex:CreateIntentVersion
	lex:CreateResourcePolicy
	lex:CreateSlotTypeVersion
	lex:CreateTestSetDiscrepancyReport
	lex:CreateUploadUrl
	lex:DeleteBot
	lex:DeleteBotAnalyzerRecommendation
	lex:DeleteBotChannelAssociation
	lex:DeleteBotReplica
	lex:DeleteExport
	lex:DeleteImport
	lex:DeleteIntentVersion
	lex:DeleteResourcePolicy

Prefixo do serviço	Ações do
	lex:DeleteSlotTypeVersion
	lex:DeleteTestSet
	lex:DeleteUtterances
	lex:DescribeBotAlias
	lex:DescribeBotAnalyzerRecommendation
	lex:DescribeBotRecommendation
	lex:DescribeBotReplica
	lex:DescribeBotResourceGeneration
	lex:DescribeBotVersion
	lex:DescribeCustomVocabularyMetadata
	lex:DescribeExport
	lex:DescribeImport
	lex:DescribeResourcePolicy
	lex:DescribeTestExecution
	lex:DescribeTestSet
	lex:DescribeTestSetDiscrepancyReport
	lex:DescribeTestSetGeneration
	lex:GenerateBotElement
	lex:GetBot
	lex:GetBotAlias
	lex:GetBotAliases

Prefixo do serviço	Ações do
	lex:GetBotChannelAssociation
	lex:GetBotChannelAssociations
	lex:GetBotVersions
	lex:GetBots
	lex:GetBuiltinIntent
	lex:GetBuiltinIntents
	lex:GetBuiltinSlotTypes
	lex:GetExport
	lex:GetImport
	lex:GetIntent
	lex:GetIntentVersions
	lex:GetIntents
	lex:GetMigration
	lex:GetMigrations
	lex:GetSlotType
	lex:GetSlotTypeVersions
	lex:GetSlotTypes
	lex:GetTestExecutionArtifactsUrl
	lex:GetUtterancesView
	lex:ListBotAliasReplicas
	lex:ListBotAliases

Prefixo do serviço	Ações do
	lex:ListBotRecommendations
	lex:ListBotReplicas
	lex:ListBotResourceGenerations
	lex:ListBotVersionReplicas
	lex:ListBotVersions
	lex:ListBots
	lex:ListBuiltInIntents
	lex:ListBuiltInSlotTypes
	lex:ListCustomVocabularyItems
	lex:ListExports
	lex:ListImports
	lex:ListIntentMetrics
	lex:ListIntentPaths
	lex:ListRecommendedIntents
	lex:ListSessionAnalyticsData
	lex:ListSessionMetrics
	lex:ListTestExecutionResultItems
	lex:ListTestExecutions
	lex:ListTestSets
	lex:PutBot
	lex:PutBotAlias

Prefixo do serviço	Ações do
	lex:PutIntent
	lex:PutSlotType
	lex:SearchAssociatedTranscripts
	lex:StartBotAnalyzer
	lex:StartBotRecommendation
	lex:StartImport
	lex:StartMigration
	lex:StartTestExecution
	lex:StartTestSetGeneration
	lex:StopBotAnalyzer
	lex:StopBotRecommendation
	lex:UpdateBotAlias
	lex:UpdateBotRecommendation
	lex:UpdateExport
	lex:UpdateResourcePolicy

Prefixo do serviço	Ações do
license-manager-linux-subscriptions	license-manager-linux-subscriptions:DeregisterSubscriptionProvider license-manager-linux-subscriptions:GetRegisteredSubscriptionProvider license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:ListRegisteredSubscriptionProviders license-manager-linux-subscriptions:RegisterSubscriptionProvider license-manager-linux-subscriptions:UpdateServiceSettings

Prefixo do serviço	Ações do
lightsail	lightsail:AllocateStaticIp
	lightsail:AttachCertificateToDistribution
	lightsail:AttachDisk
	lightsail:AttachInstancesToLoadBalancer
	lightsail:AttachLoadBalancerTlsCertificate
	lightsail:AttachStaticIp
	lightsail:CloseInstancePublicPorts
	lightsail:CopySnapshot
	lightsail:CreateBucket
	lightsail:CreateBucketAccessKey
	lightsail:CreateCertificate
	lightsail:CreateCloudFormationStack
	lightsail:CreateContactMethod
	lightsail:CreateContainerService
	lightsail:CreateContainerServiceDeployment
	lightsail:CreateContainerServiceRegistryLogin
	lightsail:CreateDisk
	lightsail:CreateDiskFromSnapshot
	lightsail:CreateDiskSnapshot
	lightsail:CreateDistribution
	lightsail:CreateDomain

Prefixo do serviço	Ações do
	lightsail:CreateGUISessionAccessDetails
	lightsail:CreateInstanceSnapshot
	lightsail:CreateInstances
	lightsail:CreateInstancesFromSnapshot
	lightsail:CreateKeyPair
	lightsail:CreateLoadBalancer
	lightsail:CreateLoadBalancerTlsCertificate
	lightsail:CreateRelationalDatabase
	lightsail:CreateRelationalDatabaseFromSnapshot
	lightsail:CreateRelationalDatabaseSnapshot
	lightsail:DeleteAlarm
	lightsail:DeleteAutoSnapshot
	lightsail:DeleteBucket
	lightsail:DeleteBucketAccessKey
	lightsail:DeleteCertificate
	lightsail:DeleteContactMethod
	lightsail:DeleteContainerImage
	lightsail:DeleteContainerService
	lightsail:DeleteDisk
	lightsail:DeleteDiskSnapshot
	lightsail:DeleteDistribution

Prefixo do serviço	Ações do
	lightsail:DeleteDomain
	lightsail:DeleteDomainEntry
	lightsail:DeleteInstance
	lightsail:DeleteInstanceSnapshot
	lightsail:DeleteKeyPair
	lightsail:DeleteKnownHostKeys
	lightsail:DeleteLoadBalancer
	lightsail:DeleteLoadBalancerTlsCertificate
	lightsail:DeleteRelationalDatabase
	lightsail:DeleteRelationalDatabaseSnapshot
	lightsail:DetachCertificateFromDistribution
	lightsail:DetachDisk
	lightsail:DetachInstancesFromLoadBalancer
	lightsail:DetachStaticIp
	lightsail:DisableAddOn
	lightsail:DownloadDefaultKeyPair
	lightsail:EnableAddOn
	lightsail:ExportSnapshot
	lightsail:GetActiveNames
	lightsail:GetAlarms
	lightsail:GetAutoSnapshots

Prefixo do serviço	Ações do
	lightsail:GetBlueprints
	lightsail:GetBucketAccessKeys
	lightsail:GetBucketBundles
	lightsail:GetBucketMetricData
	lightsail:GetBuckets
	lightsail:GetBundles
	lightsail:GetCertificates
	lightsail:GetCloudFormationStackRecords
	lightsail:GetContactMethods
	lightsail:GetContainerAPIMetadata
	lightsail:GetContainerImages
	lightsail:GetContainerLog
	lightsail:GetContainerServiceDeployments
	lightsail:GetContainerServiceMetricData
	lightsail:GetContainerServicePowers
	lightsail:GetContainerServices
	lightsail:GetCostEstimate
	lightsail:GetDisk
	lightsail:GetDiskSnapshot
	lightsail:GetDiskSnapshots
	lightsail:GetDisks

Prefixo do serviço	Ações do
	lightsail:GetDistributionBundles
	lightsail:GetDistributionLatestCacheReset
	lightsail:GetDistributionMetricData
	lightsail:GetDistributions
	lightsail:GetDomain
	lightsail:GetExportSnapshotRecords
	lightsail:GetInstance
	lightsail:GetInstanceMetricData
	lightsail:GetInstancePortStates
	lightsail:GetInstanceSnapshot
	lightsail:GetInstanceSnapshots
	lightsail:GetInstanceState
	lightsail:GetInstances
	lightsail:GetKeyPair
	lightsail:GetKeyPairs
	lightsail:GetLoadBalancer
	lightsail:GetLoadBalancerMetricData
	lightsail:GetLoadBalancerTlsCertificates
	lightsail:GetLoadBalancerTlsPolicies
	lightsail:GetLoadBalancers
	lightsail:GetOperation

Prefixo do serviço	Ações do
	lightsail:GetOperations
	lightsail:GetOperationsForResource
	lightsail:GetRegions
	lightsail:GetRelationalDatabase
	lightsail:GetRelationalDatabaseBlueprints
	lightsail:GetRelationalDatabaseBundles
	lightsail:GetRelationalDatabaseEvents
	lightsail:GetRelationalDatabaseLogEvents
	lightsail:GetRelationalDatabaseLogStreams
	lightsail:GetRelationalDatabaseMasterUserPassword
	lightsail:GetRelationalDatabaseMetricData
	lightsail:GetRelationalDatabaseParameters
	lightsail:GetRelationalDatabaseSnapshot
	lightsail:GetRelationalDatabaseSnapshots
	lightsail:GetRelationalDatabases
	lightsail:GetSetupHistory
	lightsail:GetStaticIp
	lightsail:GetStaticIps
	lightsail:ImportKeyPair
	lightsail:IsVpcPeered
	lightsail:OpenInstancePublicPorts

Prefixo do serviço	Ações do
	lightsail:PeerVpc
	lightsail:PutAlarm
	lightsail:PutInstancePublicPorts
	lightsail:RebootInstance
	lightsail:RebootRelationalDatabase
	lightsail:RegisterContainerImage
	lightsail:ReleaseStaticIp
	lightsail:ResetDistributionCache
	lightsail:SendContactMethodVerification
	lightsail:SetIpAddressType
	lightsail:SetResourceAccessForBucket
	lightsail:SetupInstanceHttps
	lightsail:StartGUISession
	lightsail:StartInstance
	lightsail:StartRelationalDatabase
	lightsail:StopGUISession
	lightsail:StopInstance
	lightsail:StopRelationalDatabase
	lightsail:TestAlarm
	lightsail:UnpeerVpc
	lightsail:UpdateBucket

Prefixo do serviço	Ações do
	lightsail:UpdateBucketBundle
	lightsail:UpdateContainerService
	lightsail:UpdateDistribution
	lightsail:UpdateDistributionBundle
	lightsail:UpdateDomainEntry
	lightsail:UpdateInstanceMetadataOptions
	lightsail:UpdateLoadBalancerAttribute
	lightsail:UpdateRelationalDatabase
	lightsail:UpdateRelationalDatabaseParameters

Prefixo do serviço	Ações do
logs	logs:AssociateKmsKey logs:AssociateSourceToS3TableIntegration logs:CancelExportTask logs:CancelImportTask logs:CreateDelivery logs:CreateExportTask logs:CreateLogAnomalyDetector logs:CreateLogGroup logs:CreateLogStream logs:CreateLookupTable logs:DeleteDataProtectionPolicy logs:DeleteDelivery logs:DeleteDeliveryDestination logs:DeleteDeliveryDestinationPolicy logs:DeleteDeliverySource logs:DeleteDestination logs:DeleteIndexPolicy logs:DeleteIntegration logs:DeleteLogAnomalyDetector logs:DeleteLogGroup logs:DeleteLogStream

Prefixo do serviço	Ações do
	logs:DeleteLookupTable logs:DeleteMetricFilter logs:DeleteQueryDefinition logs:DeleteResourcePolicy logs:DeleteRetentionPolicy logs:DeleteScheduledQuery logs:DeleteSubscriptionFilter logs:DeleteSyslogConfiguration logs:DeleteTransformer logs:DescribeAccountPolicies logs:DescribeConfigurationTemplates logs:DescribeDeliveries logs:DescribeDeliveryDestinations logs:DescribeDeliverySources logs:DescribeDestinations logs:DescribeExportTasks logs:DescribeFieldIndexes logs:DescribeImportTaskBatches logs:DescribeImportTasks logs:DescribeIndexPolicies logs:DescribeLogGroups

Prefixo do serviço	Ações do
	logs:DescribeLogStreams logs:DescribeLookupTables logs:DescribeMetricFilters logs:DescribeQueries logs:DescribeQueryDefinitions logs:DescribeResourcePolicies logs:DescribeSubscriptionFilters logs:DisassociateKmsKey logs:DisassociateSourceFromS3TableIntegration logs:GetDataProtectionPolicy logs:GetDelivery logs:GetDeliveryDestination logs:GetDeliveryDestinationPolicy logs:GetDeliverySource logs:GetIntegration logs:GetLogAnomalyDetector logs:GetLogFields logs:GetLogGroupFields logs:GetLogRecord logs:GetLookupTable logs:GetQueryResults

Prefixo do serviço	Ações do
	logs:GetScheduledQuery logs:GetScheduledQueryHistory logs:GetStorageTierPolicy logs:GetTransformer logs:ListAnomalies logs:ListIntegrations logs:ListLogAnomalyDetectors logs:ListLogGroupsForQuery logs:ListScheduledQueries logs:ListSourcesForS3TableIntegration logs:ListSyslogConfigurations logs:PutBearerTokenAuthentication logs:PutDataProtectionPolicy logs:PutDeliveryDestination logs:PutDeliveryDestinationPolicy logs:PutDeliverySource logs:PutDestination logs:PutDestinationPolicy logs:PutIndexPolicy logs:PutIntegration logs:PutLogGroupDeletionProtection

Prefixo do serviço	Ações do
	logs:PutMetricFilter logs:putQueryDefinition logs:PutResourcePolicy logs:PutRetentionPolicy logs:PutStorageTierPolicy logs:PutSubscriptionFilter logs:PutSyslogConfiguration logs:PutTransformer logs:StartLiveTail logs:StartQuery logs:StopQuery logs:TestMetricFilter logs:TestTransformer logs:UpdateAnomaly logs:UpdateDeliveryConfiguration logs:UpdateLogAnomalyDetector logs:UpdateLookupTable

Prefixo do serviço	Ações do
lookoutequipment	lookoutequipment:CreateDataset
	lookoutequipment:CreateInferenceScheduler
	lookoutequipment:CreateLabel
	lookoutequipment:CreateLabelGroup
	lookoutequipment:CreateModel
	lookoutequipment:DeleteDataset
	lookoutequipment:DeleteInferenceScheduler
	lookoutequipment:DeleteLabel
	lookoutequipment:DeleteLabelGroup
	lookoutequipment:DeleteModel
	lookoutequipment:DeleteResourcePolicy
	lookoutequipment:DeleteRetrainingScheduler
	lookoutequipment:DescribeDataIngestionJob
	lookoutequipment:DescribeDataset
	lookoutequipment:DescribeInferenceScheduler
	lookoutequipment:DescribeLabelGroup
	lookoutequipment:DescribeModel
	lookoutequipment:DescribeModelVersion
	lookoutequipment:DescribeResourcePolicy
	lookoutequipment:DescribeRetrainingScheduler
	lookoutequipment:Describelabel

Prefixo do serviço	Ações do
	lookoutequipment:ImportDataset
	lookoutequipment:ImportModelVersion
	lookoutequipment:ListDataIngestionJobs
	lookoutequipment:ListDatasets
	lookoutequipment:ListInferenceEvents
	lookoutequipment:ListInferenceExecutions
	lookoutequipment:ListInferenceSchedulers
	lookoutequipment:ListLabelGroups
	lookoutequipment:ListLabels
	lookoutequipment:ListModelVersions
	lookoutequipment:ListModels
	lookoutequipment:ListRetrainingSchedulers
	lookoutequipment:ListSensorStatistics
	lookoutequipment:PutResourcePolicy
	lookoutequipment:StartDataIngestionJob
	lookoutequipment:StartInferenceScheduler
	lookoutequipment:StartRetrainingScheduler
	lookoutequipment:StopInferenceScheduler
	lookoutequipment:StopRetrainingScheduler
	lookoutequipment:UpdateActiveModelVersion
	lookoutequipment:UpdateInferenceScheduler

Prefixo do serviço	Ações do
	lookoutequipment:UpdateLabelGroup lookoutequipment:UpdateModel lookoutequipment:UpdateRetrainingScheduler

Prefixo do serviço	Ações do
lookoutmetrics	lookoutmetrics:ActivateAnomalyDetector lookoutmetrics:BackTestAnomalyDetector lookoutmetrics:CreateAlert lookoutmetrics:CreateAnomalyDetector lookoutmetrics:CreateMetricSet lookoutmetrics:DeactivateAnomalyDetector lookoutmetrics:DeleteAlert lookoutmetrics:DeleteAnomalyDetector lookoutmetrics:DescribeAlert lookoutmetrics:DescribeAnomalyDetectionExecutions lookoutmetrics:DescribeAnomalyDetector lookoutmetrics:DescribeMetricSet lookoutmetrics:DetectMetricSetConfig lookoutmetrics:GetAnomalyGroup lookoutmetrics:GetDataQualityMetrics lookoutmetrics:GetFeedback lookoutmetrics:GetSampleData lookoutmetrics:ListAlerts lookoutmetrics:ListAnomalyDetectors lookoutmetrics:ListAnomalyGroupRelatedMetrics lookoutmetrics:ListAnomalyGroupSummaries

Prefixo do serviço	Ações do
	lookoutmetrics:ListAnomalyGroupTimeSeries lookoutmetrics:ListMetricSets lookoutmetrics:PutFeedback lookoutmetrics:UpdateAlert lookoutmetrics:UpdateAnomalyDetector lookoutmetrics:UpdateMetricSet

Prefixo do serviço	Ações do
m2	m2:CancelBatchJobExecution
	m2:CreateApplication
	m2:CreateDataSetExportTask
	m2:CreateDataSetImportTask
	m2:CreateDeployment
	m2:CreateEnvironment
	m2:DeleteApplication
	m2:DeleteApplicationFromEnvironment
	m2:DeleteEnvironment
	m2:GetApplication
	m2:GetApplicationVersion
	m2:GetBatchJobExecution
	m2:GetDataSetDetails
	m2:GetDataSetExportTask
	m2:GetDataSetImportTask
	m2:GetDeployment
	m2:GetEnvironment
	m2:GetSignedBluinsightsUrl
	m2:ListApplicationVersions
	m2:ListApplications
	m2:ListBatchJobDefinitions

Prefixo do serviço	Ações do
	m2:ListBatchJobExecutions
	m2:ListBatchJobRestartPoints
	m2:ListDataSetExportHistory
	m2:ListDataSetImportHistory
	m2:ListDataSets
	m2:ListDeployments
	m2:ListEngineVersions
	m2:ListEnvironments
	m2:StartApplication
	m2:StartBatchJob
	m2:StopApplication
	m2:UpdateApplication
	m2:UpdateEnvironment

Prefixo do serviço	Ações do
managedblockchain	managedblockchain:CreateAccessor managedblockchain:CreateMember managedblockchain:CreateNetwork managedblockchain:CreateNode managedblockchain:CreateProposal managedblockchain:DeleteAccessor managedblockchain:DeleteMember managedblockchain:DeleteNode managedblockchain:GetAccessor managedblockchain:GetMember managedblockchain:GetNetwork managedblockchain:GetNode managedblockchain:GetProposal managedblockchain:InvokeRpcPolygonMainnet managedblockchain:InvokeRpcPolygonMumbaiTestnet managedblockchain:ListAccessors managedblockchain:ListInvitations managedblockchain:ListMembers managedblockchain:ListNetworks managedblockchain:ListNodes managedblockchain:ListProposalVotes

Prefixo do serviço	Ações do
	managedblockchain:ListProposals managedblockchain:RejectInvitation managedblockchain:UpdateMember managedblockchain:UpdateNode managedblockchain:VoteOnProposal

Prefixo do serviço	Ações do
mediaconnect	mediaconnect:AddBridgeOutputs
	mediaconnect:AddBridgeSources
	mediaconnect:AddFlowMediaStreams
	mediaconnect:AddFlowOutputs
	mediaconnect:AddFlowSources
	mediaconnect:AddFlowVpcInterfaces
	mediaconnect:CreateBridge
	mediaconnect:CreateFlow
	mediaconnect:CreateGateway
	mediaconnect:DeleteBridge
	mediaconnect:DeleteFlow
	mediaconnect:DeleteGateway
	mediaconnect:DeleteRouterInput
	mediaconnect:DeleteRouterNetworkInterface
	mediaconnect:DeleteRouterOutput
	mediaconnect:DeregisterGatewayInstance
	mediaconnect:DescribeBridge
	mediaconnect:DescribeFlow
	mediaconnect:DescribeFlowSourceMetadata
	mediaconnect:DescribeFlowSourceThumbnail
	mediaconnect:DescribeGateway

Prefixo do serviço	Ações do
	mediaconnect:DescribeGatewayInstance
	mediaconnect:DescribeOffering
	mediaconnect:DescribeReservation
	mediaconnect:GetRouterInput
	mediaconnect:GetRouterInputSourceMetadata
	mediaconnect:GetRouterInputThumbnail
	mediaconnect:GetRouterNetworkInterface
	mediaconnect:GetRouterOutput
	mediaconnect:GrantFlowEntitlements
	mediaconnect:ListBridges
	mediaconnect:ListEntitlements
	mediaconnect:ListFlows
	mediaconnect:ListGatewayInstances
	mediaconnect:ListGateways
	mediaconnect:ListOfferings
	mediaconnect:ListReservations
	mediaconnect:ListRouterInputs
	mediaconnect:ListRouterNetworkInterfaces
	mediaconnect:ListRouterOutputs
	mediaconnect:PurchaseOffering
	mediaconnect:RemoveBridgeOutput

Prefixo do serviço	Ações do
	mediaconnect:RemoveBridgeSource
	mediaconnect:RemoveFlowMediaStream
	mediaconnect:RemoveFlowOutput
	mediaconnect:RemoveFlowSource
	mediaconnect:RemoveFlowVpcInterface
	mediaconnect:RestartRouterInput
	mediaconnect:RestartRouterOutput
	mediaconnect:RevokeFlowEntitlement
	mediaconnect:StartFlow
	mediaconnect:StartRouterInput
	mediaconnect:StartRouterOutput
	mediaconnect:StopFlow
	mediaconnect:StopRouterInput
	mediaconnect:StopRouterOutput
	mediaconnect:TakeRouterInput
	mediaconnect:UpdateBridge
	mediaconnect:UpdateBridgeOutput
	mediaconnect:UpdateBridgeSource
	mediaconnect:UpdateBridgeState
	mediaconnect:UpdateFlow
	mediaconnect:UpdateFlowEntitlement

Prefixo do serviço	Ações do
	mediaconnect:UpdateFlowMediaStream mediaconnect:UpdateGatewayInstance

Prefixo do serviço	Ações do
mediaconvert	mediaconvert:AssociateCertificate mediaconvert:CancelJob mediaconvert:CreateJob mediaconvert:CreateJobTemplate mediaconvert:CreatePreset mediaconvert:CreateQueue mediaconvert:CreateResourceShare mediaconvert:DeleteJobTemplate mediaconvert:DeletePolicy mediaconvert:DeletePreset mediaconvert:DeleteQueue mediaconvert:DescribeEndpoints mediaconvert:DisassociateCertificate mediaconvert:GetJob mediaconvert:GetJobTemplate mediaconvert:GetPolicy mediaconvert:GetPreset mediaconvert:GetQueue mediaconvert:ListJobTemplates mediaconvert:ListJobs mediaconvert:ListPresets

Prefixo do serviço	Ações do
	mediaconvert:ListQueues mediaconvert:ListVersions mediaconvert:Probe mediaconvert:PutPolicy mediaconvert:SearchJobs mediaconvert:UpdateJobTemplate mediaconvert:UpdatePreset mediaconvert:UpdateQueue

Prefixo do serviço	Ações do
medialive	medialive:AcceptInputDeviceTransfer
	medialive:BatchDelete
	medialive:BatchStart
	medialive:BatchStop
	medialive:BatchUpdateSchedule
	medialive:CancelInputDeviceTransfer
	medialive:ClaimDevice
	medialive:CreateChannel
	medialive:CreateChannelPlacementGroup
	medialive:CreateCloudWatchAlarmTemplate
	medialive:CreateCloudWatchAlarmTemplateGroup
	medialive:CreateCluster
	medialive:CreateEventBridgeRuleTemplate
	medialive:CreateEventBridgeRuleTemplateGroup
	medialive:CreateInput
	medialive:CreateInputSecurityGroup
	medialive:CreateMultiplex
	medialive:CreateMultiplexProgram
	medialive:CreateNetwork
	medialive:CreateNode
	medialive:CreateNodeRegistrationScript

Prefixo do serviço	Ações do
	medialive:CreatePartnerInput
	medialive:CreateSdiSource
	medialive:CreateSignalMap
	medialive:DeleteChannel
	medialive:DeleteChannelPlacementGroup
	medialive:DeleteCloudWatchAlarmTemplate
	medialive:DeleteCloudWatchAlarmTemplateGroup
	medialive:DeleteCluster
	medialive:DeleteEventBridgeRuleTemplate
	medialive:DeleteEventBridgeRuleTemplateGroup
	medialive:DeleteInput
	medialive:DeleteInputSecurityGroup
	medialive:DeleteMultiplex
	medialive:DeleteMultiplexProgram
	medialive:DeleteNetwork
	medialive:DeleteNode
	medialive:DeleteReservation
	medialive:DeleteSchedule
	medialive:DeleteSdiSource
	medialive:DeleteSignalMap
	medialive:DescribeAccountConfiguration

Prefixo do serviço	Ações do
	medialive:DescribeChannel
	medialive:DescribeChannelPlacementGroup
	medialive:DescribeCluster
	medialive:DescribeInput
	medialive:DescribeInputDevice
	medialive:DescribeInputDeviceThumbnail
	medialive:DescribeInputSecurityGroup
	medialive:DescribeMultiplex
	medialive:DescribeMultiplexProgram
	medialive:DescribeNetwork
	medialive:DescribeNode
	medialive:DescribeOffering
	medialive:DescribeReservation
	medialive:DescribeSchedule
	medialive:DescribeSdiSource
	medialive:DescribeThumbnails
	medialive:GetCloudWatchAlarmTemplate
	medialive:GetCloudWatchAlarmTemplateGroup
	medialive:GetEventBridgeRuleTemplate
	medialive:GetEventBridgeRuleTemplateGroup
	medialive:GetSignalMap

Prefixo do serviço	Ações do
	medialive:ListAlerts
	medialive:ListChannelPlacementGroups
	medialive:ListChannels
	medialive:ListCloudWatchAlarmTemplateGroups
	medialive:ListCloudWatchAlarmTemplates
	medialive:ListClusterAlerts
	medialive:ListClusters
	medialive:ListEventBridgeRuleTemplateGroups
	medialive:ListEventBridgeRuleTemplates
	medialive:ListInputDeviceTransfers
	medialive:ListInputDevices
	medialive:ListInputSecurityGroups
	medialive:ListInputs
	medialive:ListMultiplexAlerts
	medialive:ListMultiplexPrograms
	medialive:ListMultiplexes
	medialive:ListNetworks
	medialive:ListNodes
	medialive:ListOfferings
	medialive:ListReservations
	medialive:ListSdiSources

Prefixo do serviço	Ações do
	medialive:ListSignalMaps
	medialive:ListVersions
	medialive:PurchaseOffering
	medialive:RebootInputDevice
	medialive:RejectInputDeviceTransfer
	medialive:RestartChannelPipelines
	medialive:StartChannel
	medialive:StartDeleteMonitorDeployment
	medialive:StartInputDevice
	medialive:StartInputDeviceMaintenanceWindow
	medialive:StartMonitorDeployment
	medialive:StartMultiplex
	medialive:StartUpdateSignalMap
	medialive:StopChannel
	medialive:StopInputDevice
	medialive:StopMultiplex
	medialive:TransferInputDevice
	medialive:UpdateAccountConfiguration
	medialive:UpdateChannel
	medialive:UpdateChannelClass
	medialive:UpdateChannelPlacementGroup

Prefixo do serviço	Ações do
	medialive:UpdateCloudWatchAlarmTemplate
	medialive:UpdateCloudWatchAlarmTemplateGroup
	medialive:UpdateCluster
	medialive:UpdateEventBridgeRuleTemplate
	medialive:UpdateEventBridgeRuleTemplateGroup
	medialive:UpdateInput
	medialive:UpdateInputDevice
	medialive:UpdateInputSecurityGroup
	medialive:UpdateMultiplex
	medialive:UpdateMultiplexProgram
	medialive:UpdateNetwork
	medialive:UpdateNode
	medialive:UpdateNodeState
	medialive:UpdateReservation
	medialive:UpdateSdiSource

Prefixo do serviço	Ações do
mediatailor	mediatailor:ConfigureLogsForPlaybackConfiguration mediatailor:CreateChannel mediatailor:CreateLiveSource mediatailor:CreatePrefetchSchedule mediatailor:CreateProgram mediatailor:CreateSourceLocation mediatailor:CreateVodSource mediatailor:DeleteChannel mediatailor:DeleteChannelPolicy mediatailor:DeleteLiveSource mediatailor:DeletePlaybackConfiguration mediatailor:DeletePrefetchSchedule mediatailor:DeleteProgram mediatailor:DeleteSourceLocation mediatailor:DeleteVodSource mediatailor:DescribeChannel mediatailor:DescribeLiveSource mediatailor:DescribeProgram mediatailor:DescribeSourceLocation mediatailor:DescribeVodSource mediatailor:GetChannelPolicy

Prefixo do serviço	Ações do
	mediatailor:GetChannelSchedule mediatailor:GetPlaybackConfiguration mediatailor:GetPrefetchSchedule mediatailor:ListAlerts mediatailor:ListChannels mediatailor:ListLiveSources mediatailor:ListPlaybackConfigurations mediatailor:ListPrefetchSchedules mediatailor:ListSourceLocations mediatailor:ListVodSources mediatailor:PutChannelPolicy mediatailor:PutPlaybackConfiguration mediatailor:StartChannel mediatailor:StopChannel mediatailor:UpdateChannel mediatailor:UpdateLiveSource mediatailor:UpdateProgram mediatailor:UpdateSourceLocation mediatailor:UpdateVodSource

Prefixo do serviço	Ações do
memorydb	memorydb:BatchUpdateCluster
	memorydb:CopySnapshot
	memorydb:CreateAcl
	memorydb:CreateCluster
	memorydb:CreateMultiRegionCluster
	memorydb:CreateParameterGroup
	memorydb:CreateSnapshot
	memorydb:CreateSubnetGroup
	memorydb:CreateUser
	memorydb:DeleteAcl
	memorydb:DeleteCluster
	memorydb:DeleteMultiRegionCluster
	memorydb:DeleteParameterGroup
	memorydb:DeleteSnapshot
	memorydb:DeleteSubnetGroup
	memorydb:DeleteUser
	memorydb:DescribeAcls
	memorydb:DescribeClusters
	memorydb:DescribeEngineVersions
	memorydb:DescribeEvents
	memorydb:DescribeMultiRegionClusters

Prefixo do serviço	Ações do
	memorydb:DescribeMultiRegionParameterGroups
	memorydb:DescribeMultiRegionParameters
	memorydb:DescribeParameterGroups
	memorydb:DescribeParameters
	memorydb:DescribeReservedNodes
	memorydb:DescribeReservedNodesOfferings
	memorydb:DescribeServiceUpdates
	memorydb:DescribeSnapshots
	memorydb:DescribeSubnetGroups
	memorydb:DescribeUsers
	memorydb:FailoverShard
	memorydb:ListAllowedMultiRegionClusterUpdates
	memorydb:ListAllowedNodeTypeUpdates
	memorydb:PurchaseReservedNodesOffering
	memorydb:ResetParameterGroup
	memorydb:UpdateAcl
	memorydb:UpdateCluster
	memorydb:UpdateMultiRegionCluster
	memorydb:UpdateParameterGroup
	memorydb:UpdateSubnetGroup
	memorydb:UpdateUser

Prefixo do serviço	Ações do
mgh	mgh:AssociateCreatedArtifact mgh:AssociateDiscoveredResource mgh:AssociateSourceResource mgh:CreateHomeRegionControl mgh:CreateProgressUpdateStream mgh:DeleteHomeRegionControl mgh:DeleteProgressUpdateStream mgh:DescribeApplicationState mgh:DescribeHomeRegionControls mgh:DescribeMigrationTask mgh:DisassociateCreatedArtifact mgh:DisassociateDiscoveredResource mgh:DisassociateSourceResource mgh:GetHomeRegion mgh:ImportMigrationTask mgh:ListApplicationStates mgh:ListCreatedArtifacts mgh:ListDiscoveredResources mgh:ListMigrationTaskUpdates mgh:ListMigrationTasks mgh:ListProgressUpdateStreams

Prefixo do serviço	Ações do
	mgh:ListSourceResources mgh:NotifyApplicationState mgh:NotifyMigrationTaskState mgh:PutResourceAttributes

Prefixo do serviço	Ações do
mgn	mgn:ArchiveApplication
	mgn:ArchiveWave
	mgn:AssociateApplications
	mgn:AssociateSourceServers
	mgn:ChangeServerLifeCycleState
	mgn:CreateApplication
	mgn:CreateConnector
	mgn:CreateLaunchConfigurationTemplate
	mgn:CreateNetworkMigrationDefinition
	mgn:CreateReplicationConfigurationTemplate
	mgn:CreateWave
	mgn:DeleteApplication
	mgn:DeleteConnector
	mgn:DeleteJob
	mgn:DeleteLaunchConfigurationTemplate
	mgn:DeleteNetworkMigrationDefinition
	mgn:DeleteReplicationConfigurationTemplate
	mgn:DeleteSourceServer
	mgn:DeleteVcenterClient
	mgn:DeleteWave
	mgn:DescribeJobLogItems

Prefixo do serviço	Ações do
	mgn:DescribeJobs
	mgn:DescribeLaunchConfigurationTemplates
	mgn:DescribeReplicationConfigurationTemplates
	mgn:DescribeVcenterClients
	mgn:DisassociateApplications
	mgn:DisassociateSourceServers
	mgn:DisconnectFromService
	mgn:FinalizeCutover
	mgn:GetNetworkMigrationDefinition
	mgn:GetNetworkMigrationMapperSegmentConstruct
	mgn:GetReplicationConfiguration
	mgn:InitializeService
	mgn:ListConnectors
	mgn:ListExportErrors
	mgn:ListExports
	mgn:ListImportErrors
	mgn:ListImportFileEnrichments
	mgn:ListImports
	mgn:ListManagedAccounts
	mgn:ListNetworkMigrationAnalyses
	mgn:ListNetworkMigrationAnalysisResults

Prefixo do serviço	Ações do
	mgn:ListNetworkMigrationCodeGenerationSegments
	mgn:ListNetworkMigrationCodeGenerations
	mgn:ListNetworkMigrationDefinitions
	mgn:ListNetworkMigrationDeployedStacks
	mgn:ListNetworkMigrationDeployments
	mgn:ListNetworkMigrationExecutions
	mgn:ListNetworkMigrationMapperSegmentConstructs
	mgn:ListNetworkMigrationMapperSegments
	mgn:ListNetworkMigrationMappingUpdates
	mgn:ListNetworkMigrationMappings
	mgn:ListSourceServerActions
	mgn:ListTemplateActions
	mgn:MarkAsArchived
	mgn:PauseReplication
	mgn:PutSourceServerAction
	mgn:PutTemplateAction
	mgn:RemoveSourceServerAction
	mgn:RemoveTemplateAction
	mgn:ResumeReplication
	mgn:RetryDataReplication
	mgn:StartCutover

Prefixo do serviço	Ações do
	mgn:StartExport
	mgn:StartImport
	mgn:StartImportFileEnrichment
	mgn:StartNetworkMigrationCodeGeneration
	mgn:StartNetworkMigrationMapping
	mgn:StartNetworkMigrationMappingUpdate
	mgn:StartReplication
	mgn:StartTest
	mgn:StopReplication
	mgn:TerminateTargetInstances
	mgn:UnarchiveApplication
	mgn:UnarchiveWave
	mgn:UpdateApplication
	mgn:UpdateConnector
	mgn:UpdateLaunchConfigurationTemplate
	mgn:UpdateNetworkMigrationDefinition
	mgn:UpdateNetworkMigrationMapperSegment
	mgn:UpdateReplicationConfiguration
	mgn:UpdateReplicationConfigurationTemplate
	mgn:UpdateSourceServer
	mgn:UpdateSourceServerReplicationType

Prefixo do serviço	Ações do
	mgn:UpdateWave

Prefixo do serviço	Ações do
migrationhub-strategy	migrationhub-strategy:GetAntiPattern migrationhub-strategy:GetApplicationComponentDetails migrationhub-strategy:GetApplicationComponentStrategies migrationhub-strategy:GetAssessment migrationhub-strategy:GetImportFileTask migrationhub-strategy:GetLatestAssessmentId migrationhub-strategy:GetMessage migrationhub-strategy:GetPortfolioPreferences migrationhub-strategy:GetPortfolioSummary migrationhub-strategy:GetRecommendationReportDetails migrationhub-strategy:GetServerDetails migrationhub-strategy:GetServerStrategies migrationhub-strategy:ListAnalyzableServers migrationhub-strategy:ListAntiPatterns migrationhub-strategy:ListApplicationComponents migrationhub-strategy:ListCollectors migrationhub-strategy:ListImportFileTask migrationhub-strategy:ListJarArtifacts migrationhub-strategy:ListServers migrationhub-strategy:PutLogData migrationhub-strategy:PutMetricData

Prefixo do serviço	Ações do
	migrationhub-strategy:PutPortfolioPreferences migrationhub-strategy:RegisterCollector migrationhub-strategy:SendMessage migrationhub-strategy:StartAssessment migrationhub-strategy:StartImportFileTask migrationhub-strategy:StartRecommendationReportGeneration migrationhub-strategy:StopAssessment migrationhub-strategy:UpdateApplicationComponentConfig migrationhub-strategy:UpdateCollectorConfiguration migrationhub-strategy:UpdateServerConfig

Prefixo do serviço	Ações do
mobiletargeting	mobiletargeting:CreateApp
	mobiletargeting:CreateCampaign
	mobiletargeting:CreateEmailTemplate
	mobiletargeting:CreateExportJob
	mobiletargeting:CreateImportJob
	mobiletargeting:CreateInAppTemplate
	mobiletargeting:CreateJourney
	mobiletargeting:CreatePushTemplate
	mobiletargeting:CreateRecommenderConfiguration
	mobiletargeting:CreateSegment
	mobiletargeting:CreateSmsTemplate
	mobiletargeting:CreateVoiceTemplate
	mobiletargeting:DeleteAdmChannel
	mobiletargeting:DeleteApnsChannel
	mobiletargeting:DeleteApnsSandboxChannel
	mobiletargeting:DeleteApnsVoipChannel
	mobiletargeting:DeleteApnsVoipSandboxChannel
	mobiletargeting:DeleteApp
	mobiletargeting:DeleteBaiduChannel
	mobiletargeting:DeleteCampaign
	mobiletargeting:DeleteEmailChannel

Prefixo do serviço	Ações do
	mobiletargeting:DeleteEmailTemplate
	mobiletargeting:DeleteEndpoint
	mobiletargeting:DeleteEventStream
	mobiletargeting:DeleteGcmChannel
	mobiletargeting:DeleteInAppTemplate
	mobiletargeting:DeleteJourney
	mobiletargeting:DeletePushTemplate
	mobiletargeting:DeleteRecommenderConfiguration
	mobiletargeting:DeleteSegment
	mobiletargeting:DeleteSmsChannel
	mobiletargeting:DeleteSmsTemplate
	mobiletargeting:DeleteUserEndpoints
	mobiletargeting:DeleteVoiceChannel
	mobiletargeting:DeleteVoiceTemplate
	mobiletargeting:GetAdmChannel
	mobiletargeting:GetApnsChannel
	mobiletargeting:GetApnsSandboxChannel
	mobiletargeting:GetApnsVoipChannel
	mobiletargeting:GetApnsVoipSandboxChannel
	mobiletargeting:GetApp
	mobiletargeting:GetApplicationDateRangeKpi

Prefixo do serviço	Ações do
	mobiletargeting:GetApplicationSettings
	mobiletargeting:GetApps
	mobiletargeting:GetBaiduChannel
	mobiletargeting:GetCampaign
	mobiletargeting:GetCampaignActivities
	mobiletargeting:GetCampaignDateRangeKpi
	mobiletargeting:GetCampaignVersion
	mobiletargeting:GetCampaignVersions
	mobiletargeting:GetCampaigns
	mobiletargeting:GetChannels
	mobiletargeting:GetEmailChannel
	mobiletargeting:GetEmailTemplate
	mobiletargeting:GetEndpoint
	mobiletargeting:GetEventStream
	mobiletargeting:GetExportJob
	mobiletargeting:GetExportJobs
	mobiletargeting:GetGcmChannel
	mobiletargeting:GetImportJob
	mobiletargeting:GetImportJobs
	mobiletargeting:GetInAppMessages
	mobiletargeting:GetInAppTemplate

Prefixo do serviço	Ações do
	mobiletargeting:GetJourney
	mobiletargeting:GetJourneyDateRangeKpi
	mobiletargeting:GetJourneyExecutionActivityMetrics
	mobiletargeting:GetJourneyExecutionMetrics
	mobiletargeting:GetJourneyRunExecutionActivityMetrics
	mobiletargeting:GetJourneyRunExecutionMetrics
	mobiletargeting:GetJourneyRuns
	mobiletargeting:GetPushTemplate
	mobiletargeting:GetRecommenderConfiguration
	mobiletargeting:GetRecommenderConfigurations
	mobiletargeting:GetSegment
	mobiletargeting:GetSegmentExportJobs
	mobiletargeting:GetSegmentImportJobs
	mobiletargeting:GetSegmentVersion
	mobiletargeting:GetSegmentVersions
	mobiletargeting:GetSegments
	mobiletargeting:GetSmsChannel
	mobiletargeting:GetSmsTemplate
	mobiletargeting:GetUserEndpoints
	mobiletargeting:GetVoiceChannel
	mobiletargeting:GetVoiceTemplate

Prefixo do serviço	Ações do
	mobiletargeting:ListJourneys
	mobiletargeting:ListTemplateVersions
	mobiletargeting:ListTemplates
	mobiletargeting:PhoneNumberValidate
	mobiletargeting:PutEventStream
	mobiletargeting:RemoveAttributes
	mobiletargeting:UpdateAdmChannel
	mobiletargeting:UpdateApnsChannel
	mobiletargeting:UpdateApnsSandboxChannel
	mobiletargeting:UpdateApnsVoipChannel
	mobiletargeting:UpdateApnsVoipSandboxChannel
	mobiletargeting:UpdateApplicationSettings
	mobiletargeting:UpdateBaiduChannel
	mobiletargeting:UpdateCampaign
	mobiletargeting:UpdateEmailChannel
	mobiletargeting:UpdateEmailTemplate
	mobiletargeting:UpdateEndpoint
	mobiletargeting:UpdateEndpointsBatch
	mobiletargeting:UpdateGcmChannel
	mobiletargeting:UpdateInAppTemplate
	mobiletargeting:UpdateJourney

Prefixo do serviço	Ações do
	mobiletargeting:UpdateJourneyState
	mobiletargeting:UpdatePushTemplate
	mobiletargeting:UpdateRecommenderConfiguration
	mobiletargeting:UpdateSegment
	mobiletargeting:UpdateSmsChannel
	mobiletargeting:UpdateSmsTemplate
	mobiletargeting:UpdateTemplateActiveVersion
	mobiletargeting:UpdateVoiceChannel
	mobiletargeting:UpdateVoiceTemplate
	mobiletargeting:VerifyOTPMessage

Prefixo do serviço	Ações do
mq	mq:CreateBroker mq:CreateConfiguration mq:CreateUser mq:DeleteBroker mq:DeleteConfiguration mq:DeleteUser mq:DescribeBroker mq:DescribeBrokerEngineTypes mq:DescribeBrokerInstanceOptions mq:DescribeConfiguration mq:DescribeConfigurationRevision mq:DescribeSharedResources mq:DescribeUser mq:ListBrokers mq:ListConfigurationRevisions mq:ListConfigurations mq:ListUsers mq:Promote mq:RebootBroker mq:UpdateBroker mq:UpdateConfiguration

Prefixo do serviço	Ações do
	mq:UpdateUser

Prefixo do serviço	Ações do
networkmanager	networkmanager:AcceptAttachment
	networkmanager:AssociateConnectPeer
	networkmanager:AssociateCustomerGateway
	networkmanager:AssociateLink
	networkmanager:AssociateTransitGatewayConnectPeer
	networkmanager:CreateConnectAttachment
	networkmanager:CreateConnectPeer
	networkmanager:CreateConnection
	networkmanager:CreateCoreNetwork
	networkmanager:CreateDevice
	networkmanager:CreateDirectConnectGatewayAttachment
	networkmanager:CreateGlobalNetwork
	networkmanager:CreateLink
	networkmanager:CreateSite
	networkmanager:CreateSiteToSiteVpnAttachment
	networkmanager:CreateTransitGatewayPeering
	networkmanager:CreateTransitGatewayRouteTableAttachment
	networkmanager:CreateVpcAttachment
	networkmanager:DeleteAttachment
	networkmanager:DeleteConnectPeer
	networkmanager:DeleteConnection

Prefixo do serviço	Ações do
	networkmanager:DeleteCoreNetwork
	networkmanager:DeleteCoreNetworkPolicyVersion
	networkmanager:DeleteDevice
	networkmanager:DeleteGlobalNetwork
	networkmanager:DeleteLink
	networkmanager:DeletePeering
	networkmanager:DeleteResourcePolicy
	networkmanager:DeleteSite
	networkmanager:DeregisterTransitGateway
	networkmanager:DescribeGlobalNetworks
	networkmanager:DisassociateConnectPeer
	networkmanager:DisassociateCustomerGateway
	networkmanager:DisassociateLink
	networkmanager:DisassociateTransitGatewayConnectPeer
	networkmanager:ExecuteCoreNetworkChangeSet
	networkmanager:GetConnectAttachment
	networkmanager:GetConnectPeer
	networkmanager:GetConnectPeerAssociations
	networkmanager:GetConnections
	networkmanager:GetCoreNetwork
	networkmanager:GetCoreNetworkChangeEvents

Prefixo do serviço	Ações do
	networkmanager:GetCoreNetworkChangeSet
	networkmanager:GetCoreNetworkPolicy
	networkmanager:GetCustomerGatewayAssociations
	networkmanager:GetDevices
	networkmanager:GetLinkAssociations
	networkmanager:GetLinks
	networkmanager:GetNetworkResourceCounts
	networkmanager:GetNetworkResourceRelationships
	networkmanager:GetNetworkResources
	networkmanager:GetNetworkRoutes
	networkmanager:GetNetworkTelemetry
	networkmanager:GetResourcePolicy
	networkmanager:GetRouteAnalysis
	networkmanager:GetSiteToSiteVpnAttachment
	networkmanager:GetSites
	networkmanager:GetTransitGatewayConnectPeerAssociations
	networkmanager:GetTransitGatewayPeering
	networkmanager:GetTransitGatewayRegistrations
	networkmanager:GetTransitGatewayRouteTableAttachment
	networkmanager:GetVpcAttachment
	networkmanager:ListAttachmentRoutingPolicyAssociations

Prefixo do serviço	Ações do
	networkmanager:ListAttachments
	networkmanager:ListConnectPeers
	networkmanager:ListCoreNetworkPolicyVersions
	networkmanager:ListCoreNetworkPrefixListAssociations
	networkmanager:ListCoreNetworkRoutingInformation
	networkmanager:ListCoreNetworks
	networkmanager:ListOrganizationServiceAccessStatus
	networkmanager:ListPeerings
	networkmanager:PutAttachmentRoutingPolicyLabel
	networkmanager:PutCoreNetworkPolicy
	networkmanager:PutResourcePolicy
	networkmanager:RegisterTransitGateway
	networkmanager:RejectAttachment
	networkmanager:RemoveAttachmentRoutingPolicyLabel
	networkmanager:RestoreCoreNetworkPolicyVersion
	networkmanager:StartOrganizationServiceAccessUpdate
	networkmanager:StartRouteAnalysis
	networkmanager:UpdateConnection
	networkmanager:UpdateCoreNetwork
	networkmanager:UpdateDevice
	networkmanager:UpdateDirectConnectGatewayAttachment

Prefixo do serviço	Ações do
	networkmanager:UpdateGlobalNetwork
	networkmanager:UpdateLink
	networkmanager:UpdateNetworkResourceMetadata
	networkmanager:UpdateSite
	networkmanager:UpdateVpcAttachment

Prefixo do serviço	Ações do
nimble	nimble:AcceptEulas nimble:CreateLaunchProfile nimble:CreateStreamingImage nimble:CreateStreamingSession nimble:CreateStreamingSessionStream nimble:CreateStudio nimble:CreateStudioComponent nimble:DeleteLaunchProfile nimble:DeleteLaunchProfileMember nimble:DeleteStreamingImage nimble:DeleteStreamingSession nimble:DeleteStudio nimble:DeleteStudioComponent nimble:DeleteStudioMember nimble:GetEula nimble:GetLaunchProfileDetails nimble:GetStreamingImage nimble:GetStreamingSession nimble:GetStreamingSessionBackup nimble:GetStreamingSessionStream nimble:GetStudio

Prefixo do serviço	Ações do
	nimble:GetStudioComponent nimble:GetStudioMember nimble:ListEulas nimble:ListLaunchProfileMembers nimble:ListLaunchProfiles nimble:ListStreamingImages nimble:ListStreamingSessionBackups nimble:ListStreamingSessions nimble:ListStudioComponents nimble:ListStudioMembers nimble:ListStudios nimble:PutLaunchProfileMembers nimble:PutStudioMembers nimble:StartStreamingSession nimble:StartStudioSSOConfigurationRepair nimble:StopStreamingSession nimble:UpdateLaunchProfile nimble:UpdateLaunchProfileMember nimble:UpdateStreamingImage nimble:UpdateStudio nimble:UpdateStudioComponent

Prefixo do serviço	Ações do
omics	omics:AbortMultipartReadSetUpload
	omics:AcceptShare
	omics:BatchDeleteReadSet
	omics:CancelAnnotationImportJob
	omics:CancelRun
	omics:CancelRunBatch
	omics:CancelVariantImportJob
	omics:CompleteMultipartReadSetUpload
	omics:CreateAnnotationStore
	omics:CreateAnnotationStoreVersion
	omics:CreateConfiguration
	omics:CreateMultipartReadSetUpload
	omics:CreateReferenceStore
	omics:CreateRunGroup
	omics:CreateSequenceStore
	omics:CreateShare
	omics:CreateVariantStore
	omics:CreateWorkflow
	omics:CreateWorkflowVersion
	omics:DeleteAnnotationStore
	omics:DeleteAnnotationStoreVersions

Prefixo do serviço	Ações do
	omics:DeleteBatch
	omics:DeleteConfiguration
	omics:DeleteReference
	omics:DeleteReferenceStore
	omics:DeleteRun
	omics:DeleteRunBatch
	omics:DeleteRunGroup
	omics:DeleteSequenceStore
	omics:DeleteShare
	omics:DeleteVariantStore
	omics:DeleteWorkflow
	omics:DeleteWorkflowVersion
	omics:GetAnnotationImportJob
	omics:GetAnnotationStore
	omics:GetAnnotationStoreVersion
	omics:GetBatch
	omics:GetConfiguration
	omics:GetReadSet
	omics:GetReadSetActivationJob
	omics:GetReadSetExportJob
	omics:GetReadSetImportJob

Prefixo do serviço	Ações do
	omics:GetReadSetMetadata
	omics:GetReference
	omics:GetReferenceImportJob
	omics:GetReferenceMetadata
	omics:GetReferenceStore
	omics:GetRun
	omics:GetRunGroup
	omics:GetRunTask
	omics:GetSequenceStore
	omics:GetShare
	omics:GetVariantImportJob
	omics:GetVariantStore
	omics:GetWorkflow
	omics:GetWorkflowVersion
	omics:ListAnnotationImportJobs
	omics:ListAnnotationStoreVersions
	omics:ListAnnotationStores
	omics:ListBatch
	omics:ListConfigurations
	omics:ListMultipartReadSetUploads
	omics:ListReadSetActivationJobs

Prefixo do serviço	Ações do
	omics:ListReadSetExportJobs
	omics:ListReadSetImportJobs
	omics:ListReadSetUploadParts
	omics:ListReadSets
	omics:ListReferenceImportJobs
	omics:ListReferenceStores
	omics:ListReferences
	omics:ListRunGroups
	omics:ListRunTasks
	omics:ListRuns
	omics:ListRunsInBatch
	omics:ListSequenceStores
	omics:ListShares
	omics:ListVariantImportJobs
	omics:ListVariantStores
	omics:ListWorkflowVersions
	omics:ListWorkflows
	omics:StartAnnotationImportJob
	omics:StartReadSetActivationJob
	omics:StartReadSetExportJob
	omics:StartReadSetImportJob

Prefixo do serviço	Ações do
	omics:StartReferenceImportJob
	omics:StartRun
	omics:StartRunBatch
	omics:StartVariantImportJob
	omics:UpdateAnnotationStore
	omics:UpdateAnnotationStoreVersion
	omics:UpdateRunGroup
	omics:UpdateVariantStore
	omics:UpdateWorkflow
	omics:UpdateWorkflowVersion
	omics:UploadReadSetPart

Prefixo do serviço	Ações do
opsworks	opsworks:AssignInstance opsworks:AssignVolume opsworks:AssociateElasticIp opsworks:AttachElasticLoadBalancer opsworks:CloneStack opsworks:CreateApp opsworks:CreateDeployment opsworks:CreateInstance opsworks:CreateLayer opsworks:CreateStack opsworks:CreateUserProfile opsworks:DeleteApp opsworks:DeleteInstance opsworks:DeleteLayer opsworks:DeleteStack opsworks:DeleteUserProfile opsworks:DeregisterEcsCluster opsworks:DeregisterElasticIp opsworks:DeregisterInstance opsworks:DeregisterRdsDbInstance opsworks:DeregisterVolume

Prefixo do serviço	Ações do
	opsworks:DescribeAgentVersions opsworks:DescribeApps opsworks:DescribeCommands opsworks:DescribeDeployments opsworks:DescribeEcsClusters opsworks:DescribeElasticIps opsworks:DescribeElasticLoadBalancers opsworks:DescribeInstances opsworks:DescribeLayers opsworks:DescribeLoadBasedAutoScaling opsworks:DescribeMyUserProfile opsworks:DescribeOperatingSystems opsworks:DescribePermissions opsworks:DescribeRaidArrays opsworks:DescribeRdsDbInstances opsworks:DescribeServiceErrors opsworks:DescribeStackProvisioningParameters opsworks:DescribeStackSummary opsworks:DescribeStacks opsworks:DescribeTimeBasedAutoScaling opsworks:DescribeUserProfiles

Prefixo do serviço	Ações do
	opsworks:DescribeVolumes opsworks:DetachElasticLoadBalancer opsworks:DisassociateElasticIp opsworks:GetHostnameSuggestion opsworks:GrantAccess opsworks:RebootInstance opsworks:RegisterEcsCluster opsworks:RegisterElasticIp opsworks:RegisterInstance opsworks:RegisterRdsDbInstance opsworks:RegisterVolume opsworks:SetLoadBasedAutoScaling opsworks:SetPermission opsworks:SetTimeBasedAutoScaling opsworks:StartInstance opsworks:StartStack opsworks:StopInstance opsworks:StopStack opsworks:UnassignInstance opsworks:UnassignVolume opsworks:UpdateApp

Prefixo do serviço	Ações do
	opsworks:UpdateElasticIp opsworks:UpdateInstance opsworks:UpdateLayer opsworks:UpdateMyUserProfile opsworks:UpdateRdsDbInstance opsworks:UpdateStack opsworks:UpdateUserProfile opsworks:UpdateVolume

Prefixo do serviço	Ações do
opsworks-cm	opsworks-cm:AssociateNode
	opsworks-cm:CreateBackup
	opsworks-cm:CreateServer
	opsworks-cm:DeleteBackup
	opsworks-cm:DeleteServer
	opsworks-cm:DescribeAccountAttributes
	opsworks-cm:DescribeBackups
	opsworks-cm:DescribeEvents
	opsworks-cm:DescribeNodeAssociationStatus
	opsworks-cm:DescribeServers
	opsworks-cm:DisassociateNode
	opsworks-cm:ExportServerEngineAttribute
	opsworks-cm:RestoreServer
	opsworks-cm:StartMaintenance
	opsworks-cm:UpdateServer
	opsworks-cm:UpdateServerEngineAttributes

Prefixo do serviço	Ações do
organizações	organizations:AcceptHandshake organizations:AttachPolicy organizations:CancelHandshake organizations:CloseAccount organizations:CreateAccount organizations:CreateGovCloudAccount organizations:CreateOrganization organizations:CreateOrganizationalUnit organizations:CreatePolicy organizations:DeclineHandshake organizations:DeleteOrganization organizations:DeleteOrganizationalUnit organizations:DeletePolicy organizations:DeleteResourcePolicy organizations:DeregisterDelegatedAdministrator organizations:DescribeAccount organizations:DescribeCreateAccountStatus organizations:DescribeEffectivePolicy organizations:DescribeHandshake organizations:DescribeOrganization organizations:DescribeOrganizationalUnit

Prefixo do serviço	Ações do
	organizations:DescribePolicy organizations:DescribeResourcePolicy organizations:DescribeResponsibilityTransfer organizations:DetachPolicy organizations:DisableAWSServiceAccess organizations:DisablePolicyType organizations:EnableAWSServiceAccess organizations:EnableAllFeatures organizations:EnablePolicyType organizations:InviteAccountToOrganization organizations:LeaveOrganization organizations:ListAWSServiceAccessForOrganization organizations:ListAccounts organizations:ListAccountsForParent organizations:ListAccountsWithInvalidEffectivePolicy organizations:ListChildren organizations:ListCreateAccountStatus organizations:ListDelegatedAdministrators organizations:ListDelegatedServicesForAccount organizations:ListHandshakesForAccount organizations:ListHandshakesForOrganization

Prefixo do serviço	Ações do
	organizations:ListInboundResponsibilityTransfers organizations:ListOrganizationalUnitsForParent organizations:ListOutboundResponsibilityTransfers organizations:ListParents organizations:ListPolicies organizations:ListPoliciesForTarget organizations:ListRoots organizations:ListTargetsForPolicy organizations:MoveAccount organizations:PutResourcePolicy organizations:RegisterDelegatedAdministrator organizations:RemoveAccountFromOrganization organizations:TerminateResponsibilityTransfer organizations:UpdateOrganizationalUnit organizations:UpdatePolicy organizations:UpdateResponsibilityTransfer

Prefixo do serviço	Ações do
outposts	outposts:CancelCapacityTask
	outposts:CancelOrder
	outposts:CreateOrder
	outposts:CreateOutpost
	outposts:CreatePrivateConnectivityConfig
	outposts:CreateQuote
	outposts:CreateRenewal
	outposts:CreateSite
	outposts:DeleteOutpost
	outposts:DeleteQuote
	outposts:DeleteSite
	outposts:GetCapacityTask
	outposts:GetCatalogItem
	outposts:GetConnection
	outposts:GetOrder
	outposts:GetOutpost
	outposts:GetOutpostBillingInformation
	outposts:GetOutpostInstanceTypes
	outposts:GetOutpostSupportedInstanceTypes
	outposts:GetPrivateConnectivityConfig
	outposts:GetQuote

Prefixo do serviço	Ações do
	outposts:GetRenewalPricing
	outposts:GetSite
	outposts:GetSiteAddress
	outposts:ListAssetInstances
	outposts:ListAssets
	outposts:ListBlockingInstancesForCapacityTask
	outposts:ListCapacityTasks
	outposts:ListCatalogItems
	outposts:ListOrderableInstanceTypes
	outposts:ListOrders
	outposts:ListOutposts
	outposts:ListQuotes
	outposts:ListSites
	outposts:StartCapacityTask
	outposts:StartConnection
	outposts:UpdateOutpost
	outposts:UpdateQuote
	outposts:UpdateSite
	outposts:UpdateSiteAddress
	outposts:UpdateSiteRackPhysicalProperties

Prefixo do serviço	Ações do
pi	pi:CreatePerformanceAnalysisReport pi:DeletePerformanceAnalysisReport pi:DescribeDimensionKeys pi:GetDimensionKeyDetails pi:GetPerformanceAnalysisReport pi:GetResourceMetadata pi:GetResourceMetrics pi:ListAvailableResourceDimensions pi:ListAvailableResourceMetrics pi:ListPerformanceAnalysisReports
pipes	pipes:CreatePipe pipes:DeletePipe pipes:DescribePipe pipes:ListPipes pipes:StartPipe pipes:StopPipe pipes:UpdatePipe

Prefixo do serviço	Ações do
polly	polly:DeleteLexicon polly:DescribeVoices polly:GetLexicon polly:GetSpeechSynthesisTask polly:ListLexicons polly:ListSpeechSynthesisTasks polly:PutLexicon polly:StartSpeechSynthesisTask polly:SynthesizeSpeech

Prefixo do serviço	Ações do
perfil	profile:AddProfileKey
	profile:BatchGetCalculatedAttributeForProfile
	profile:BatchGetProfile
	profile:CreateCalculatedAttributeDefinition
	profile:CreateDomain
	profile:CreateEventStream
	profile:CreateProfile
	profile:CreateRecommender
	profile:CreateRecommenderFilter
	profile:CreateRecommenderSchema
	profile:CreateSegmentDefinition
	profile:CreateSegmentEstimate
	profile:CreateSegmentSnapshot
	profile:CreateUploadJob
	profile:DeleteCalculatedAttributeDefinition
	profile:DeleteDomain
	profile:DeleteDomainObjectType
	profile:DeleteEventStream
	profile:DeleteIntegration
	profile:DeleteProfile
	profile:DeleteProfileKey

Prefixo do serviço	Ações do
	profile:DeleteProfileObject
	profile:DeleteProfileObjectType
	profile:DeleteRecommender
	profile:DeleteRecommenderFilter
	profile:DeleteRecommenderSchema
	profile:DeleteSegmentDefinition
	profile:DeleteWorkflow
	profile:DetectProfileObjectType
	profile:GetAutoMergingPreview
	profile:GetCalculatedAttributeDefinition
	profile:GetCalculatedAttributeForProfile
	profile:GetDomain
	profile:GetDomainObjectType
	profile:GetEventStream
	profile:GetIdentityResolutionJob
	profile:GetIntegration
	profile:GetMatches
	profile:GetObjectTypeAttributeStatistics
	profile:GetProfileObjectType
	profile:GetProfileObjectTypeTemplate
	profile:GetProfileRecommendations

Prefixo do serviço	Ações do
	profile:GetRecommender
	profile:GetRecommenderFilter
	profile:GetRecommenderSchema
	profile:GetSegmentDefinition
	profile:GetSegmentEstimate
	profile:GetSegmentMembership
	profile:GetSegmentSnapshot
	profile:GetSimilarProfiles
	profile:GetUploadJob
	profile:GetUploadJobPath
	profile:GetWorkflow
	profile:GetWorkflowSteps
	profile:ListAccountIntegrations
	profile:ListCalculatedAttributeDefinitions
	profile:ListCalculatedAttributesForProfile
	profile:ListDomainLayouts
	profile:ListDomainObjectTypes
	profile:ListDomains
	profile:ListEventStreams
	profile:ListIdentityResolutionJobs
	profile:ListIntegrations

Prefixo do serviço	Ações do
	profile:ListObjectTypeAttributeValues
	profile:ListObjectTypeAttributes
	profile:ListProfileAttributeValues
	profile:ListProfileObjectTypeTemplates
	profile:ListProfileObjectTypes
	profile:ListProfileObjects
	profile:ListRecommenderFilters
	profile:ListRecommenderRecipes
	profile:ListRecommenderSchemas
	profile:ListRecommenders
	profile:ListRuleBasedMatches
	profile:ListSegmentDefinitions
	profile:ListUploadJobs
	profile:ListWorkflows
	profile:MergeProfiles
	profile:PutDomainObjectType
	profile:PutIntegration
	profile:PutProfileObject
	profile:PutProfileObjectType
	profile:SearchProfiles
	profile:StartRecommender

Prefixo do serviço	Ações do
	profile:StartUploadJob profile:StopRecommender profile:StopUploadJob profile:UpdateCalculatedAttributeDefinition profile:UpdateDomain profile:UpdateProfile profile:UpdateRecommender

Prefixo do serviço	Ações do
qldb	qldb:CancelJournalKinesisStream qldb:CreateLedger qldb:DeleteLedger qldb:DescribeJournalKinesisStream qldb:DescribeJournalS3Export qldb:DescribeLedger qldb:ExportJournalToS3 qldb:GetBlock qldb:GetDigest qldb:GetRevision qldb:ListJournalKinesisStreamsForLedger qldb:ListJournalS3Exports qldb:ListJournalS3ExportsForLedger qldb:ListLedgers qldb:StreamJournalToKinesis qldb:UpdateLedger qldb:UpdateLedgerPermissionsMode

Prefixo do serviço	Ações do
ram	ram:AcceptResourceShareInvitation
	ram:AssociateResourceShare
	ram:AssociateResourceSharePermission
	ram:CreatePermission
	ram:CreatePermissionVersion
	ram:CreateResourceShare
	ram:DeletePermission
	ram:DeletePermissionVersion
	ram:DeleteResourceShare
	ram:DisassociateResourceShare
	ram:DisassociateResourceSharePermission
	ram:EnableSharingWithAwsOrganization
	ram:GetPermission
	ram:GetResourcePolicies
	ram:GetResourceShareAssociations
	ram:GetResourceShareInvitations
	ram:GetResourceShares
	ram:ListPendingInvitationResources
	ram:ListPermissionAssociations
	ram:ListPermissionVersions
	ram:ListPermissions

Prefixo do serviço	Ações do
	ram:ListPrincipals ram:ListReplacePermissionAssociationsWork ram:ListResourceSharePermissions ram:ListResourceTypes ram:ListResources ram:ListSourceAssociations ram:PromotePermissionCreatedFromPolicy ram:PromoteResourceShareCreatedFromPolicy ram:RejectResourceShareInvitation ram:ReplacePermissionAssociations ram:SetDefaultPermissionVersion ram:UpdateResourceShare
rbin	rbin:CreateRule rbin:DeleteRule rbin:GetRule rbin:ListRules rbin:LockRule rbin:UnlockRule rbin:UpdateRule

Prefixo do serviço	Ações do
rds	rds:AddRoleToDBCluster
	rds:AddRoleToDBInstance
	rds:AddSourceIdentifierToSubscription
	rds:ApplyPendingMaintenanceAction
	rds:AuthorizeDBSecurityGroupIngress
	rds:BacktrackDBCluster
	rds:CancelExportTask
	rds:CopyDBClusterParameterGroup
	rds:CopyDBClusterSnapshot
	rds:CopyDBParameterGroup
	rds:CopyDBSnapshot
	rds:CopyOptionGroup
	rds:CreateCustomDBEngineVersion
	rds:CreateDBClusterParameterGroup
	rds:CreateDBParameterGroup
	rds:CreateDBProxy
	rds:CreateDBProxyEndpoint
	rds:CreateDBSecurityGroup
	rds:CreateDBSubnetGroup
	rds:CreateEventSubscription
	rds:CreateGlobalCluster

Prefixo do serviço	Ações do
	rds:CreateOptionGroup
	rds:DeleteBlueGreenDeployment
	rds:DeleteDBClusterAutomatedBackup
	rds:DeleteDBClusterParameterGroup
	rds:DeleteDBClusterSnapshot
	rds:DeleteDBInstanceAutomatedBackup
	rds:DeleteDBParameterGroup
	rds:DeleteDBProxy
	rds:DeleteDBProxyEndpoint
	rds:DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rds:DeleteDBSubnetGroup
	rds:DeleteEventSubscription
	rds:DeleteGlobalCluster
	rds:DeleteOptionGroup
	rds:DeregisterDBProxyTargets
	rds:DescribeAccountAttributes
	rds:DescribeBlueGreenDeployments
	rds:DescribeCertificates
	rds:DescribeDBClusterAutomatedBackups
	rds:DescribeDBClusterBacktracks

Prefixo do serviço	Ações do
	rds:DescribeDBClusterEndpoints
	rds:DescribeDBClusterParameterGroups
	rds:DescribeDBClusterParameters
	rds:DescribeDBClusterSnapshotAttributes
	rds:DescribeDBClusterSnapshots
	rds:DescribeDBClusters
	rds:DescribeDBEngineVersions
	rds:DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rds:DescribeDBLogFiles
	rds:DescribeDBMajorEngineVersions
	rds:DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds:DescribeDBProxyEndpoints
	rds:DescribeDBProxyTargetGroups
	rds:DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rds:DescribeDBSecurityGroups
	rds:DescribeDBSnapshotAttributes
	rds:DescribeDBSnapshotTenantDatabases

Prefixo do serviço	Ações do
	rds:DescribeDBSnapshots
	rds:DescribeDBSubnetGroups
	rds:DescribeEngineDefaultClusterParameters
	rds:DescribeEngineDefaultParameters
	rds:DescribeEventCategories
	rds:DescribeEventSubscriptions
	rds:DescribeEvents
	rds:DescribeExportTasks
	rds:DescribeGlobalClusters
	rds:DescribeIntegrations
	rds:DescribeOptionGroupOptions
	rds:DescribeOptionGroups
	rds:DescribeOrderableDBInstanceOptions
	rds:DescribePendingMaintenanceActions
	rds:DescribeReservedDBInstances
	rds:DescribeReservedDBInstancesOfferings
	rds:DescribeSourceRegions
	rds:DescribeTenantDatabases
	rds:DescribeValidDBInstanceModifications
	rds:DownloadCompleteDBLogFile
	rds:DownloadDBLogFilePortion

Prefixo do serviço	Ações do
	rds:FailoverDBCluster
	rds:FailoverGlobalCluster
	rds:ModifyActivityStream
	rds:ModifyCertificates
	rds:ModifyCurrentDBClusterCapacity
	rds:ModifyDBClusterEndpoint
	rds:ModifyDBClusterParameterGroup
	rds:ModifyDBClusterSnapshotAttribute
	rds:ModifyDBParameterGroup
	rds:ModifyDBProxy
	rds:ModifyDBProxyEndpoint
	rds:ModifyDBProxyTargetGroup
	rds:ModifyDBRecommendation
	rds:ModifyDBSnapshot
	rds:ModifyDBSnapshotAttribute
	rds:ModifyDBSubnetGroup
	rds:ModifyEventSubscription
	rds:ModifyGlobalCluster
	rds:ModifyOptionGroup
	rds:ModifyTenantDatabase
	rds:PurchaseReservedDBInstancesOffering

Prefixo do serviço	Ações do
	rds:RebootDBCluster
	rds:RegisterDBProxyTargets
	rds:RemoveFromGlobalCluster
	rds:RemoveRoleFromDBCluster
	rds:RemoveRoleFromDBInstance
	rds:RemoveSourceIdentifierFromSubscription
	rds:ResetDBClusterParameterGroup
	rds:ResetDBParameterGroup
	rds:RestoreDBClusterFromS3
	rds:RestoreDBClusterFromSnapshot
	rds:RestoreDBClusterToPointInTime
	rds:RestoreDBInstanceFromDBSnapshot
	rds:RestoreDBInstanceFromS3
	rds:RestoreDBInstanceToPointInTime
	rds:RevokeDBSecurityGroupIngress
	rds:StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rds:StartDBInstanceAutomatedBackupsReplication
	rds:StartExportTask
	rds:StopActivityStream

Prefixo do serviço	Ações do
	rds:StopDBCluster rds:StopDBInstance rds:StopDBInstanceAutomatedBackupsReplication rds:SwitchoverBlueGreenDeployment rds:SwitchoverGlobalCluster rds:SwitchoverReadReplica

Prefixo do serviço	Ações do
redshift	redshift:AcceptReservedNodeExchange redshift:AddPartner redshift:AssociateDataShareConsumer redshift:AuthorizeClusterSecurityGroupIngress redshift:AuthorizeDataShare redshift:AuthorizeEndpointAccess redshift:AuthorizeSnapshotAccess redshift:BatchDeleteClusterSnapshots redshift:BatchModifyClusterSnapshots redshift:CancelResize redshift:CopyClusterSnapshot redshift:CreateAuthenticationProfile redshift:CreateCluster redshift:CreateClusterParameterGroup redshift:CreateClusterSecurityGroup redshift:CreateClusterSnapshot redshift:CreateClusterSubnetGroup redshift:CreateCustomDomainAssociation redshift:CreateEndpointAccess redshift:CreateEventSubscription redshift:CreateHsmClientCertificate

Prefixo do serviço	Ações do
	redshift:CreateHsmConfiguration redshift:CreateIntegration redshift:CreateRedshiftIdcApplication redshift:CreateScheduledAction redshift:CreateSnapshotCopyGrant redshift:CreateSnapshotSchedule redshift:CreateUsageLimit redshift:DeauthorizeDataShare redshift:DeleteAuthenticationProfile redshift:DeleteCluster redshift:DeleteClusterParameterGroup redshift:DeleteClusterSecurityGroup redshift:DeleteClusterSnapshot redshift:DeleteClusterSubnetGroup redshift:DeleteCustomDomainAssociation redshift:DeleteEndpointAccess redshift:DeleteEventSubscription redshift:DeleteHsmClientCertificate redshift:DeleteHsmConfiguration redshift:DeletePartner redshift:DeleteRedshiftIdcApplication

Prefixo do serviço	Ações do
	redshift:DeleteResourcePolicy redshift:DeleteScheduledAction redshift:DeleteSnapshotCopyGrant redshift:DeleteSnapshotSchedule redshift:DeleteUsageLimit redshift:DeregisterNamespace redshift:DescribeAccountAttributes redshift:DescribeAuthenticationProfiles redshift:DescribeClusterDbRevisions redshift:DescribeClusterParameterGroups redshift:DescribeClusterParameters redshift:DescribeClusterSecurityGroups redshift:DescribeClusterSnapshots redshift:DescribeClusterSubnetGroups redshift:DescribeClusterTracks redshift:DescribeClusterVersions redshift:DescribeClusters redshift:DescribeCustomDomainAssociations redshift:DescribeDataShares redshift:DescribeDataSharesForConsumer redshift:DescribeDataSharesForProducer

Prefixo do serviço	Ações do
	redshift:DescribeDefaultClusterParameters redshift:DescribeEndpointAccess redshift:DescribeEndpointAuthorization redshift:DescribeEventCategories redshift:DescribeEventSubscriptions redshift:DescribeEvents redshift:DescribeHsmClientCertificates redshift:DescribeHsmConfigurations redshift:DescribeInboundIntegrations redshift:DescribeIntegrations redshift:DescribeLoggingStatus redshift:DescribeNodeConfigurationOptions redshift:DescribeOrderableClusterOptions redshift:DescribePartners redshift:DescribeRedshiftIdcApplications redshift:DescribeReservedNodeExchangeStatus redshift:DescribeReservedNodeOfferings redshift:DescribeReservedNodes redshift:DescribeResize redshift:DescribeScheduledActions redshift:DescribeSnapshotCopyGrants

Prefixo do serviço	Ações do
	redshift:DescribeSnapshotSchedules redshift:DescribeStorage redshift:DescribeTableRestoreStatus redshift:DescribeUsageLimits redshift:DisableLogging redshift:DisableSnapshotCopy redshift:DisassociateDataShareConsumer redshift:EnableLogging redshift:EnableSnapshotCopy redshift:FailoverPrimaryCompute redshift:GetClusterCredentials redshift:GetClusterCredentialsWithIAM redshift:GetIdentityCenterAuthToken redshift:GetReservedNodeExchangeConfigurationOptions redshift:GetReservedNodeExchangeOfferings redshift:GetResourcePolicy redshift:ListRecommendations redshift:ModifyAquaConfiguration redshift:ModifyAuthenticationProfile redshift:ModifyCluster redshift:ModifyClusterDbRevision

Prefixo do serviço	Ações do
	redshift:ModifyClusterIamRoles redshift:ModifyClusterMaintenance redshift:ModifyClusterParameterGroup redshift:ModifyClusterSnapshot redshift:ModifyClusterSnapshotSchedule redshift:ModifyClusterSubnetGroup redshift:ModifyCustomDomainAssociation redshift:ModifyEndpointAccess redshift:ModifyEventSubscription redshift:ModifyRedshiftIdcApplication redshift:ModifyScheduledAction redshift:ModifySnapshotCopyRetentionPeriod redshift:ModifySnapshotSchedule redshift:ModifyUsageLimit redshift:PauseCluster redshift:PurchaseReservedNodeOffering redshift:PutResourcePolicy redshift:RebootCluster redshift:RegisterNamespace redshift:RejectDataShare redshift:ResetClusterParameterGroup

Prefixo do serviço	Ações do
	redshift:ResizeCluster redshift:RestoreFromClusterSnapshot redshift:RestoreTableFromClusterSnapshot redshift:ResumeCluster redshift:RevokeClusterSecurityGroupIngress redshift:RevokeEndpointAccess redshift:RevokeSnapshotAccess redshift:RotateEncryptionKey redshift:UpdatePartnerStatus
redshift-data	redshift-data:BatchExecuteStatement redshift-data:CancelStatement redshift-data:DescribeStatement redshift-data:DescribeTable redshift-data:ExecuteStatement redshift-data:GetStatementResult redshift-data:ListDatabases redshift-data:ListSchemas redshift-data:ListSessions redshift-data:ListStatements redshift-data:ListTables

Prefixo do serviço	Ações do
refactor-spaces	refactor-spaces:CreateApplication refactor-spaces:CreateEnvironment refactor-spaces:CreateRoute refactor-spaces:CreateService refactor-spaces:DeleteApplication refactor-spaces:DeleteEnvironment refactor-spaces:DeleteResourcePolicy refactor-spaces:DeleteRoute refactor-spaces:DeleteService refactor-spaces:GetApplication refactor-spaces:GetEnvironment refactor-spaces:GetResourcePolicy refactor-spaces:GetRoute refactor-spaces:GetService refactor-spaces:ListApplications refactor-spaces:ListEnvironmentVpcs refactor-spaces:ListEnvironments refactor-spaces:ListRoutes refactor-spaces:ListServices refactor-spaces:PutResourcePolicy refactor-spaces:UpdateRoute

Prefixo do serviço	Ações do
rekognition	rekognition:AssociateFaces
	rekognition:CompareFaces
	rekognition:CopyProjectVersion
	rekognition:CreateCollection
	rekognition:CreateDataset
	rekognition:CreateFaceLivenessSession
	rekognition:CreateProject
	rekognition:CreateProjectVersion
	rekognition:CreateStreamProcessor
	rekognition:CreateUser
	rekognition:DeleteCollection
	rekognition:DeleteDataset
	rekognition:DeleteFaces
	rekognition:DeleteProject
	rekognition:DeleteProjectPolicy
	rekognition:DeleteProjectVersion
	rekognition:DeleteStreamProcessor
	rekognition:DeleteUser
	rekognition:DescribeCollection
	rekognition:DescribeDataset
	rekognition:DescribeProjectVersions

Prefixo do serviço	Ações do
	rekognition:DescribeProjects
	rekognition:DescribeStreamProcessor
	rekognition:DetectCustomLabels
	rekognition:DetectFaces
	rekognition:DetectLabels
	rekognition:DetectModerationLabels
	rekognition:DetectProtectiveEquipment
	rekognition:DetectText
	rekognition:DisassociateFaces
	rekognition:DistributeDatasetEntries
	rekognition:GetCelebrityInfo
	rekognition:GetCelebrityRecognition
	rekognition:GetContentModeration
	rekognition:GetFaceDetection
	rekognition:GetFaceLivenessSessionResults
	rekognition:GetFaceSearch
	rekognition:GetLabelDetection
	rekognition:GetMediaAnalysisJob
	rekognition:GetPersonTracking
	rekognition:GetSegmentDetection
	rekognition:GetTextDetection

Prefixo do serviço	Ações do
	rekognition:IndexFaces
	rekognition:ListCollections
	rekognition:ListDatasetEntries
	rekognition:ListDatasetLabels
	rekognition:ListFaces
	rekognition:ListMediaAnalysisJobs
	rekognition:ListProjectPolicies
	rekognition:ListStreamProcessors
	rekognition:ListUsers
	rekognition:PutProjectPolicy
	rekognition:RecognizeCelebrities
	rekognition:SearchFaces
	rekognition:SearchFacesByImage
	rekognition:SearchUsers
	rekognition:SearchUsersByImage
	rekognition:StartCelebrityRecognition
	rekognition:StartContentModeration
	rekognition:StartFaceDetection
	rekognition:StartFaceLivenessSession
	rekognition:StartFaceSearch
	rekognition:StartLabelDetection

Prefixo do serviço	Ações do
	rekognition:StartMediaAnalysisJob
	rekognition:StartPersonTracking
	rekognition:StartProjectVersion
	rekognition:StartSegmentDetection
	rekognition:StartStreamProcessor
	rekognition:StartTextDetection
	rekognition:StopProjectVersion
	rekognition:StopStreamProcessor
	rekognition:UpdateDatasetEntries
	rekognition:UpdateStreamProcessor

Prefixo do serviço	Ações do
resiliencehub	resiliencehub:AcceptResourceGroupingRecommendations resiliencehub:AddDraftAppVersionResourceMappings resiliencehub:BatchUpdateRecommendationStatus resiliencehub:CreateApp resiliencehub:CreateAppVersionAppComponent resiliencehub:CreateAppVersionResource resiliencehub:CreateRecommendationTemplate resiliencehub:CreateResiliencyPolicy resiliencehub:DeleteApp resiliencehub:DeleteAppAssessment resiliencehub:DeleteAppInputSource resiliencehub:DeleteAppVersionAppComponent resiliencehub:DeleteAppVersionResource resiliencehub:DeleteRecommendationTemplate resiliencehub:DeleteResiliencyPolicy resiliencehub:DescribeApp resiliencehub:DescribeAppAssessment resiliencehub:DescribeAppVersion resiliencehub:DescribeAppVersionAppComponent resiliencehub:DescribeAppVersionResource resiliencehub:DescribeAppVersionResourcesResolutionStatus

Prefixo do serviço	Ações do
	resiliencehub:DescribeAppVersionTemplate resiliencehub:DescribeDraftAppVersionResourcesImportStatus resiliencehub:DescribeMetricsExport resiliencehub:DescribeResiliencyPolicy resiliencehub:DescribeResourceGroupingRecommendationTask resiliencehub:ImportResourcesToDraftAppVersion resiliencehub:ListAlarmRecommendations resiliencehub:ListAppAssessmentComplianceDrifts resiliencehub:ListAppAssessmentResourceDrifts resiliencehub:ListAppAssessments resiliencehub:ListAppComponentCompliances resiliencehub:ListAppComponentRecommendations resiliencehub:ListAppInputSources resiliencehub:ListAppVersionAppComponents resiliencehub:ListAppVersionResourceMappings resiliencehub:ListAppVersionResources resiliencehub:ListAppVersions resiliencehub:ListApps resiliencehub:ListMetrics resiliencehub:ListRecommendationTemplates resiliencehub:ListResiliencyPolicies

Prefixo do serviço	Ações do
	resiliencehub:ListResourceGroupingRecommendations resiliencehub:ListSopRecommendations resiliencehub:ListSuggestedResiliencyPolicies resiliencehub:ListTestRecommendations resiliencehub:ListUnsupportedAppVersionResources resiliencehub:PublishAppVersion resiliencehub:PutDraftAppVersionTemplate resiliencehub:RejectResourceGroupingRecommendations resiliencehub:RemoveDraftAppVersionResourceMappings resiliencehub:ResolveAppVersionResources resiliencehub:StartAppAssessment resiliencehub:StartResourceGroupingRecommendationTask resiliencehub:UpdateApp resiliencehub:UpdateAppVersion resiliencehub:UpdateAppVersionAppComponent resiliencehub:UpdateAppVersionResource resiliencehub:UpdateResiliencyPolicy

Prefixo do serviço	Ações do
resource-explorer-2	resource-explorer-2:AssociateDefaultView
	resource-explorer-2:BatchGetView
	resource-explorer-2:CreateIndex
	resource-explorer-2:CreateResourceExplorerSetup
	resource-explorer-2:CreateView
	resource-explorer-2:DeleteIndex
	resource-explorer-2:DeleteResourceExplorerSetup
	resource-explorer-2:DeleteView
	resource-explorer-2:DisassociateDefaultView
	resource-explorer-2:GetAccountLevelServiceConfiguration
	resource-explorer-2:GetDefaultView
	resource-explorer-2:GetIndex
	resource-explorer-2:GetManagedView
	resource-explorer-2:GetResourceExplorerSetup
	resource-explorer-2:GetServiceIndex
	resource-explorer-2:GetServiceView
	resource-explorer-2:ListIndexes
	resource-explorer-2:ListIndexesForMembers
	resource-explorer-2:ListManagedViews
	resource-explorer-2:ListServiceIndexes
	resource-explorer-2:ListServiceViews

Prefixo do serviço	Ações do
	resource-explorer-2:ListStreamingAccessForServices
	resource-explorer-2:ListSupportedResourceTypes
	resource-explorer-2:ListViews
	resource-explorer-2:Search
	resource-explorer-2:UpdateIndexType
	resource-explorer-2:UpdateView

Prefixo do serviço	Ações do
resource-groups	resource-groups:CancelTagSyncTask resource-groups:GetAccountSettings resource-groups:GetGroup resource-groups:GetGroupConfiguration resource-groups:GetGroupQuery resource-groups:GetTagSyncTask resource-groups:GroupResources resource-groups:ListGroupResources resource-groups:ListGroupingStatuses resource-groups:ListGroups resource-groups:ListTagSyncTasks resource-groups:PutGroupConfiguration resource-groups:SearchResources resource-groups:StartTagSyncTask resource-groups:UngroupResources resource-groups:UpdateAccountSettings resource-groups:UpdateGroup resource-groups:UpdateGroupQuery

Prefixo do serviço	Ações do
robomaker	robomaker:BatchDeleteWorlds
	robomaker:BatchDescribeSimulationJob
	robomaker:CancelDeploymentJob
	robomaker:CancelSimulationJob
	robomaker:CancelSimulationJobBatch
	robomaker:CancelWorldExportJob
	robomaker:CancelWorldGenerationJob
	robomaker:CreateDeploymentJob
	robomaker:CreateFleet
	robomaker:CreateRobot
	robomaker:CreateRobotApplication
	robomaker:CreateRobotApplicationVersion
	robomaker:CreateSimulationApplication
	robomaker:CreateSimulationApplicationVersion
	robomaker:CreateSimulationJob
	robomaker:CreateWorldExportJob
	robomaker:CreateWorldGenerationJob
	robomaker:CreateWorldTemplate
	robomaker:DeleteFleet
	robomaker:DeleteRobot
	robomaker:DeleteRobotApplication

Prefixo do serviço	Ações do
	robomaker:DeleteSimulationApplication
	robomaker:DeleteWorldTemplate
	robomaker:DeregisterRobot
	robomaker:DescribeDeploymentJob
	robomaker:DescribeFleet
	robomaker:DescribeRobot
	robomaker:DescribeRobotApplication
	robomaker:DescribeSimulationApplication
	robomaker:DescribeSimulationJob
	robomaker:DescribeSimulationJobBatch
	robomaker:DescribeWorld
	robomaker:DescribeWorldExportJob
	robomaker:DescribeWorldGenerationJob
	robomaker:DescribeWorldTemplate
	robomaker:GetWorldTemplateBody
	robomaker:ListDeploymentJobs
	robomaker:ListFleets
	robomaker:ListRobotApplications
	robomaker:ListRobots
	robomaker:ListSimulationApplications
	robomaker:ListSimulationJobBatches

Prefixo do serviço	Ações do
	robomaker:ListSimulationJobs
	robomaker:ListWorldExportJobs
	robomaker:ListWorldGenerationJobs
	robomaker:ListWorldTemplates
	robomaker:ListWorlds
	robomaker:RegisterRobot
	robomaker:RestartSimulationJob
	robomaker:StartSimulationJobBatch
	robomaker:SyncDeploymentJob
	robomaker:UpdateRobotApplication
	robomaker:UpdateSimulationApplication
	robomaker:UpdateWorldTemplate

Prefixo do serviço	Ações do
rolesanywhere	rolesanywhere:CreateProfile rolesanywhere:CreateTrustAnchor rolesanywhere:DeleteAttributeMapping rolesanywhere:DeleteCrl rolesanywhere:DeleteProfile rolesanywhere:DeleteTrustAnchor rolesanywhere:DisableCrl rolesanywhere:DisableProfile rolesanywhere:DisableTrustAnchor rolesanywhere:EnableCrl rolesanywhere:EnableProfile rolesanywhere:EnableTrustAnchor rolesanywhere:GetCrl rolesanywhere:GetProfile rolesanywhere:GetSubject rolesanywhere:GetTrustAnchor rolesanywhere:ImportCrl rolesanywhere:ListCrls rolesanywhere:ListProfiles rolesanywhere:ListSubjects rolesanywhere:ListTrustAnchors

Prefixo do serviço	Ações do
	rolesanywhere:PutAttributeMapping rolesanywhere:PutNotificationSettings rolesanywhere:ResetNotificationSettings rolesanywhere:UpdateCrl rolesanywhere:UpdateProfile rolesanywhere:UpdateTrustAnchor

Prefixo do serviço	Ações do
route53	route53:ActivateKeySigningKey
	route53:AssociateVPCWithHostedZone
	route53:ChangeCidrCollection
	route53:ChangeResourceRecordSets
	route53:CreateCidrCollection
	route53:CreateHealthCheck
	route53:CreateHostedZone
	route53:CreateKeySigningKey
	route53:CreateQueryLoggingConfig
	route53:CreateReusableDelegationSet
	route53:CreateTrafficPolicy
	route53:CreateTrafficPolicyInstance
	route53:CreateTrafficPolicyVersion
	route53:CreateVPCAssociationAuthorization
	route53:DeactivateKeySigningKey
	route53:DeleteCidrCollection
	route53:DeleteHealthCheck
	route53:DeleteHostedZone
	route53:DeleteKeySigningKey
	route53:DeleteQueryLoggingConfig
	route53:DeleteReusableDelegationSet

Prefixo do serviço	Ações do
	route53:DeleteTrafficPolicy
	route53:DeleteTrafficPolicyInstance
	route53:DeleteVPCAssociationAuthorization
	route53:DisableHostedZoneDNSSEC
	route53:DisassociateVPCFromHostedZone
	route53:EnableHostedZoneDNSSEC
	route53:GetAccountLimit
	route53:GetChange
	route53:GetCheckerIpRanges
	route53:GetDNSSEC
	route53:GetGeoLocation
	route53:GetHealthCheck
	route53:GetHealthCheckCount
	route53:GetHealthCheckLastFailureReason
	route53:GetHealthCheckStatus
	route53:GetHostedZone
	route53:GetHostedZoneCount
	route53:GetHostedZoneLimit
	route53:GetQueryLoggingConfig
	route53:GetReusableDelegationSet
	route53:GetReusableDelegationSetLimit

Prefixo do serviço	Ações do
	route53:GetTrafficPolicy
	route53:GetTrafficPolicyInstance
	route53:GetTrafficPolicyInstanceCount
	route53:ListCidrBlocks
	route53:ListCidrCollections
	route53:ListCidrLocations
	route53:ListGeoLocations
	route53:ListHealthChecks
	route53:ListHostedZones
	route53:ListHostedZonesByName
	route53:ListHostedZonesByVPC
	route53:ListQueryLoggingConfigs
	route53:ListResourceRecordSets
	route53:ListReusableDelegationSets
	route53:ListTrafficPolicies
	route53:ListTrafficPolicyInstances
	route53:ListTrafficPolicyInstancesByHostedZone
	route53:ListTrafficPolicyInstancesByPolicy
	route53:ListTrafficPolicyVersions
	route53:ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

Prefixo do serviço	Ações do
	route53:UpdateHealthCheck
	route53:UpdateHostedZoneComment
	route53:UpdateTrafficPolicyComment
	route53:UpdateTrafficPolicyInstance

Prefixo do serviço	Ações do
route53-recovery-control-config	route53-recovery-control-config:CreateCluster route53-recovery-control-config:CreateControlPanel route53-recovery-control-config:CreateRoutingControl route53-recovery-control-config:CreateSafetyRule route53-recovery-control-config:DeleteCluster route53-recovery-control-config:DeleteControlPanel route53-recovery-control-config:DeleteRoutingControl route53-recovery-control-config:DeleteSafetyRule route53-recovery-control-config:DescribeCluster route53-recovery-control-config:DescribeControlPanel route53-recovery-control-config:DescribeRoutingControl route53-recovery-control-config:DescribeSafetyRule route53-recovery-control-config:GetResourcePolicy route53-recovery-control-config:ListAssociatedRoute53HealthChecks route53-recovery-control-config:ListClusters route53-recovery-control-config:ListControlPanels route53-recovery-control-config:ListRoutingControls route53-recovery-control-config:ListSafetyRules route53-recovery-control-config:UpdateCluster route53-recovery-control-config:UpdateControlPanel

Prefixo do serviço	Ações do
	route53-recovery-control-config:UpdateRoutingControl route53-recovery-control-config:UpdateSafetyRule

Prefixo do serviço	Ações do
route53-recovery-readiness	route53-recovery-readiness:CreateCell route53-recovery-readiness:CreateCrossAccountAuthorization route53-recovery-readiness:CreateReadinessCheck route53-recovery-readiness:CreateRecoveryGroup route53-recovery-readiness:CreateResourceSet route53-recovery-readiness:DeleteCell route53-recovery-readiness:DeleteCrossAccountAuthorization route53-recovery-readiness:DeleteReadinessCheck route53-recovery-readiness:DeleteRecoveryGroup route53-recovery-readiness:DeleteResourceSet route53-recovery-readiness:GetArchitectureRecommendations route53-recovery-readiness:GetCell route53-recovery-readiness:GetCellReadinessSummary route53-recovery-readiness:GetReadinessCheck route53-recovery-readiness:GetReadinessCheckResourceStatus route53-recovery-readiness:GetReadinessCheckStatus route53-recovery-readiness:GetRecoveryGroup route53-recovery-readiness:GetRecoveryGroupReadinessSummary route53-recovery-readiness:GetResourceSet route53-recovery-readiness:ListCells route53-recovery-readiness:ListCrossAccountAuthorizations

Prefixo do serviço	Ações do
	route53-recovery-readiness:ListReadinessChecks
	route53-recovery-readiness:ListRecoveryGroups
	route53-recovery-readiness:ListResourceSets
	route53-recovery-readiness:ListRules
	route53-recovery-readiness:UpdateCell
	route53-recovery-readiness:UpdateReadinessCheck
	route53-recovery-readiness:UpdateRecoveryGroup
	route53-recovery-readiness:UpdateResourceSet

Prefixo do serviço	Ações do
route53resolver	route53resolver:AssociateFirewallRuleGroup route53resolver:AssociateResolverEndpointIpAddress route53resolver:AssociateResolverQueryLogConfig route53resolver:AssociateResolverRule route53resolver:CreateFirewallDomainList route53resolver:CreateFirewallRule route53resolver:CreateFirewallRuleGroup route53resolver:CreateResolverEndpoint route53resolver:CreateResolverQueryLogConfig route53resolver:CreateResolverRule route53resolver:DeleteFirewallDomainList route53resolver:DeleteFirewallRule route53resolver:DeleteFirewallRuleGroup route53resolver:DeleteOutpostResolver route53resolver:DeleteResolverEndpoint route53resolver:DeleteResolverQueryLogConfig route53resolver:DeleteResolverRule route53resolver:DisassociateFirewallRuleGroup route53resolver:DisassociateResolverEndpointIpAddress route53resolver:DisassociateResolverQueryLogConfig route53resolver:DisassociateResolverRule

Prefixo do serviço	Ações do
	route53resolver:GetFirewallConfig route53resolver:GetFirewallDomainList route53resolver:GetFirewallRuleGroup route53resolver:GetFirewallRuleGroupAssociation route53resolver:GetFirewallRuleGroupPolicy route53resolver:GetOutpostResolver route53resolver:GetResolverConfig route53resolver:GetResolverDnssecConfig route53resolver:GetResolverEndpoint route53resolver:GetResolverQueryLogConfig route53resolver:GetResolverQueryLogConfigAssociation route53resolver:GetResolverQueryLogConfigPolicy route53resolver:GetResolverRule route53resolver:GetResolverRuleAssociation route53resolver:GetResolverRulePolicy route53resolver:ImportFirewallDomains route53resolver:ListFirewallConfigs route53resolver:ListFirewallDomainLists route53resolver:ListFirewallDomains route53resolver:ListFirewallRuleGroupAssociations route53resolver:ListFirewallRuleGroups

Prefixo do serviço	Ações do
	route53resolver:ListFirewallRuleTypes route53resolver:ListFirewallRules route53resolver:ListOutpostResolvers route53resolver:ListResolverConfigs route53resolver:ListResolverDnssecConfigs route53resolver:ListResolverEndpointIpAddresses route53resolver:ListResolverEndpoints route53resolver:ListResolverQueryLogConfigAssociations route53resolver:ListResolverQueryLogConfigs route53resolver:ListResolverRuleAssociations route53resolver:ListResolverRules route53resolver:PutFirewallRuleGroupPolicy route53resolver:PutResolverQueryLogConfigPolicy route53resolver:UpdateFirewallConfig route53resolver:UpdateFirewallDomains route53resolver:UpdateFirewallRule route53resolver:UpdateFirewallRuleGroupAssociation route53resolver:UpdateOutpostResolver route53resolver:UpdateResolverConfig route53resolver:UpdateResolverDnssecConfig route53resolver:UpdateResolverEndpoint

Prefixo do serviço	Ações do
	route53resolver:UpdateResolverRule
rum	rum:BatchCreateRumMetricDefinitions
	rum:BatchDeleteRumMetricDefinitions
	rum:BatchGetRumMetricDefinitions
	rum:CreateAppMonitor
	rum:DeleteAppMonitor
	rum:DeleteResourcePolicy
	rum:DeleteRumMetricsDestination
	rum:GetAppMonitor
	rum:GetAppMonitorData
	rum:GetResourcePolicy
	rum:ListAppMonitors
	rum:ListRumMetricsDestinations
	rum:PutResourcePolicy
	rum:PutRumMetricsDestination
	rum:UpdateAppMonitor
	rum:UpdateRumMetricDefinition

Prefixo do serviço	Ações do
s3	s3:AssociateAccessGrantsIdentityCenter s3:CreateAccessGrant s3:CreateAccessGrantsInstance s3:CreateAccessGrantsLocation s3:CreateAccessPoint s3:CreateAccessPointForObjectLambda s3:CreateBucket s3:CreateBucketMetadataTableConfiguration s3:CreateJob s3:CreateMultiRegionAccessPoint s3:DeleteAccessGrant s3:DeleteAccessGrantsInstance s3:DeleteAccessGrantsInstanceResourcePolicy s3:DeleteAccessGrantsLocation s3:DeleteAccessPoint s3:DeleteAccessPointForObjectLambda s3:DeleteAccessPointPolicy s3:DeleteAccessPointPolicyForObjectLambda s3:DeleteBucket s3:DeleteBucketMetadataTableConfiguration s3:DeleteBucketPolicy

Prefixo do serviço	Ações do
	s3:DeleteBucketWebsite s3:DeleteMultiRegionAccessPoint s3:DeleteStorageLensConfiguration s3:DescribeJob s3:DescribeMultiRegionAccessPointOperation s3:DissociateAccessGrantsIdentityCenter s3:GetAccelerateConfiguration s3:GetAccessGrant s3:GetAccessGrantsInstance s3:GetAccessGrantsInstanceForPrefix s3:GetAccessGrantsInstanceResourcePolicy s3:GetAccessGrantsLocation s3:GetAccessPoint s3:GetAccessPointConfigurationForObjectLambda s3:GetAccessPointForObjectLambda s3:GetAccessPointPolicy s3:GetAccessPointPolicyForObjectLambda s3:GetAccessPointPolicyStatus s3:GetAccessPointPolicyStatusForObjectLambda s3:GetAccountPublicAccessBlock s3:GetAnalyticsConfiguration

Prefixo do serviço	Ações do
	s3:GetBucketAbac s3:GetBucketAcl s3:GetBucketCORS s3:GetBucketLocation s3:GetBucketLogging s3:GetBucketNotification s3:GetBucketObjectLockConfiguration s3:GetBucketOwnershipControls s3:GetBucketPolicy s3:GetBucketPolicyStatus s3:GetBucketPublicAccessBlock s3:GetBucketRequestPayment s3:GetBucketVersioning s3:GetBucketWebsite s3:GetDataAccess s3:GetEncryptionConfiguration s3:GetIntelligentTieringConfiguration s3:GetInventoryConfiguration s3:GetLifecycleConfiguration s3:GetMetricsConfiguration s3:GetMultiRegionAccessPoint

Prefixo do serviço	Ações do
	s3:GetMultiRegionAccessPointPolicy s3:GetMultiRegionAccessPointPolicyStatus s3:GetMultiRegionAccessPointRoutes s3:GetReplicationConfiguration s3:GetStorageLensConfiguration s3:GetStorageLensDashboard s3:ListAccessGrants s3:ListAccessGrantsInstances s3:ListAccessGrantsLocations s3:ListAccessPoints s3:ListAccessPointsForObjectLambda s3:ListAllMyBuckets s3:ListBucketMultipartUploads s3:ListCallerAccessGrants s3:ListJobs s3:ListMultiRegionAccessPoints s3:ListStorageLensConfigurations s3:PutAccelerateConfiguration s3:PutAccessGrantsInstanceResourcePolicy s3:PutAccessPointConfigurationForObjectLambda s3:PutAccessPointPolicy

Prefixo do serviço	Ações do
	s3:PutAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:PutAnalyticsConfiguration s3:PutBucketAbac s3:PutBucketAcl s3:PutBucketCORS s3:PutBucketLogging s3:PutBucketNotification s3:PutBucketObjectLockConfiguration s3:PutBucketOwnershipControls s3:PutBucketPolicy s3:PutBucketPublicAccessBlock s3:PutBucketRequestPayment s3:PutBucketVersioning s3:PutBucketWebsite s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutMetricsConfiguration s3:PutMultiRegionAccessPointPolicy

Prefixo do serviço	Ações do
	s3:PutReplicationConfiguration s3:PutStorageLensConfiguration s3:SubmitMultiRegionAccessPointRoutes s3:UpdateAccessGrantsLocation s3:UpdateBucketMetadataAnnotationTableConfiguration s3:UpdateBucketMetadataJournalTableConfiguration s3:UpdateJobPriority s3:UpdateJobStatus
s3-outposts	s3-outposts:CreateEndpoint s3-outposts:DeleteEndpoint s3-outposts:ListEndpoints s3-outposts:ListOutpostsWithS3 s3-outposts:ListSharedEndpoints

Prefixo do serviço	Ações do
sagemaker-geospatial	sagemaker-geospatial:DeleteEarthObservationJob sagemaker-geospatial:DeleteVectorEnrichmentJob sagemaker-geospatial:ExportEarthObservationJob sagemaker-geospatial:ExportVectorEnrichmentJob sagemaker-geospatial:GetEarthObservationJob sagemaker-geospatial:GetRasterDataCollection sagemaker-geospatial:GetTile sagemaker-geospatial:GetVectorEnrichmentJob sagemaker-geospatial:ListEarthObservationJobs sagemaker-geospatial:ListRasterDataCollections sagemaker-geospatial:ListVectorEnrichmentJobs sagemaker-geospatial:SearchRasterDataCollection sagemaker-geospatial:StartEarthObservationJob sagemaker-geospatial:StartVectorEnrichmentJob sagemaker-geospatial:StopEarthObservationJob sagemaker-geospatial:StopVectorEnrichmentJob

Prefixo do serviço	Ações do
savingsplans	savingsplans:CreateSavingsPlan
	savingsplans:DeleteQueuedSavingsPlan
	savingsplans:DescribeSavingsPlanRates
	savingsplans:DescribeSavingsPlans
	savingsplans:DescribeSavingsPlansOfferingRates
	savingsplans:DescribeSavingsPlansOfferings
	savingsplans:ReturnSavingsPlan

Prefixo do serviço	Ações do
schemas	schemas:CreateDiscoverer schemas:CreateRegistry schemas:CreateSchema schemas:DeleteDiscoverer schemas:DeleteRegistry schemas:DeleteResourcePolicy schemas:DeleteSchema schemas:DeleteSchemaVersion schemas:DescribeCodeBinding schemas:DescribeDiscoverer schemas:DescribeRegistry schemas:DescribeSchema schemas:ExportSchema schemas:GetCodeBindingSource schemas:GetDiscoveredSchema schemas:GetResourcePolicy schemas:ListDiscoverers schemas:ListRegistries schemas:ListSchemaVersions schemas:ListSchemas schemas:PutCodeBinding

Prefixo do serviço	Ações do
	schemas:PutResourcePolicy schemas:SearchSchemas schemas:StartDiscoverer schemas:StopDiscoverer schemas:UpdateDiscoverer schemas:UpdateRegistry schemas:UpdateSchema
sdb	sdb:CreateDomain sdb:DeleteDomain sdb:DomainMetadata sdb:ListDomains

Prefixo do serviço	Ações do
secretsmanager	secretsmanager:CancelRotateSecret secretsmanager:CreateSecret secretsmanager:DeleteResourcePolicy secretsmanager:DeleteSecret secretsmanager:DescribeSecret secretsmanager:GetRandomPassword secretsmanager:GetResourcePolicy secretsmanager:GetSecretValue secretsmanager:ListSecretVersionIds secretsmanager:ListSecrets secretsmanager:PutResourcePolicy secretsmanager:PutSecretValue secretsmanager:RemoveRegionsFromReplication secretsmanager:ReplicateSecretToRegions secretsmanager:RestoreSecret secretsmanager:RotateSecret secretsmanager:StopReplicationToReplica secretsmanager:UpdateSecret secretsmanager:ValidateResourcePolicy

Prefixo do serviço	Ações do
securityhub	securityhub:AcceptAdministratorInvitation
	securityhub:AcceptInvitation
	securityhub:BatchDeleteAutomationRules
	securityhub:BatchDisableStandards
	securityhub:BatchEnableStandards
	securityhub:BatchGetAutomationRules
	securityhub:BatchGetConfigurationPolicyAssociations
	securityhub:BatchGetEnabledRegionsV2
	securityhub:BatchGetSecurityControls
	securityhub:BatchGetStandardsControlAssociations
	securityhub:BatchImportFindings
	securityhub:BatchUpdateAutomationRules
	securityhub:BatchUpdateFindings
	securityhub:BatchUpdateStandardsControlAssociations
	securityhub:ConnectorRegistrationsV2
	securityhub:CreateActionTarget
	securityhub:CreateAggregatorV2
	securityhub:CreateAutomationRule
	securityhub:CreateAutomationRuleV2
	securityhub:CreateConfigurationPolicy
	securityhub:CreateConnector

Prefixo do serviço	Ações do
	securityhub:CreateConnectorV2
	securityhub:CreateFindingAggregator
	securityhub:CreateInsight
	securityhub:CreateMembers
	securityhub:CreateTicketV2
	securityhub:DeclineInvitations
	securityhub:DeleteActionTarget
	securityhub:DeleteAggregatorV2
	securityhub:DeleteAutomationRuleV2
	securityhub:DeleteConfigurationPolicy
	securityhub:DeleteConnector
	securityhub:DeleteConnectorV2
	securityhub:DeleteFindingAggregator
	securityhub:DeleteInsight
	securityhub:DeleteInvitations
	securityhub:DeleteMembers
	securityhub:DescribeActionTargets
	securityhub:DescribeHub
	securityhub:DescribeOrganizationConfiguration
	securityhub:DescribeProducts
	securityhub:DescribeSecurityHubV2

Prefixo do serviço	Ações do
	securityhub:DescribeStandards
	securityhub:DisableImportFindingsForProduct
	securityhub:DisableOrganizationAdminAccount
	securityhub:DisableSecurityHub
	securityhub:DisableSecurityHubFeatureV2
	securityhub:DisableSecurityHubV2
	securityhub:DisassociateFromAdministratorAccount
	securityhub:DisassociateFromMasterAccount
	securityhub:DisassociateMembers
	securityhub:EnableImportFindingsForProduct
	securityhub:EnableOrganizationAdminAccount
	securityhub:EnableSecurityHub
	securityhub:EnableSecurityHubFeatureV2
	securityhub:GetAdministratorAccount
	securityhub:GetAggregatorV2
	securityhub:GetAutomationRuleV2
	securityhub:GetConfigurationPolicy
	securityhub:GetConfigurationPolicyAssociation
	securityhub:GetConnector
	securityhub:GetConnectorV2
	securityhub:GetCoverageStatisticsV2

Prefixo do serviço	Ações do
	securityhub:GetEnabledStandards
	securityhub:GetFindingAggregator
	securityhub:GetFindingHistory
	securityhub:GetFindings
	securityhub:GetInsightResults
	securityhub:GetInsights
	securityhub:GetInvitationsCount
	securityhub:GetMasterAccount
	securityhub:GetMembers
	securityhub:GetSecurityControlDefinition
	securityhub:InviteMembers
	securityhub:ListAggregatorsV2
	securityhub:ListAutomationRules
	securityhub:ListAutomationRulesV2
	securityhub:ListConfigurationPolicies
	securityhub:ListConfigurationPolicyAssociations
	securityhub:ListConnectors
	securityhub:ListConnectorsV2
	securityhub:ListEnabledProductsForImport
	securityhub:ListFindingAggregators
	securityhub:ListInvitations

Prefixo do serviço	Ações do
	securityhub:ListMembers
	securityhub:ListOrganizationAdminAccounts
	securityhub:ListSecurityControlDefinitions
	securityhub:ListStandardsControlAssociations
	securityhub:StartConfigurationPolicyAssociation
	securityhub:StartConfigurationPolicyDisassociation
	securityhub:UpdateActionTarget
	securityhub:UpdateAggregatorV2
	securityhub:UpdateAutomationRuleV2
	securityhub:UpdateConfigurationPolicy
	securityhub:UpdateConnector
	securityhub:UpdateConnectorV2
	securityhub:UpdateFindingAggregator
	securityhub:UpdateFindings
	securityhub:UpdateInsight
	securityhub:UpdateOrganizationConfiguration
	securityhub:UpdateSecurityControl
	securityhub:UpdateSecurityHubConfiguration

Prefixo do serviço	Ações do
securitylake	securitylake:CreateAwsLogSource securitylake:CreateCustomLogSource securitylake:CreateDataLakeExceptionSubscription securitylake:CreateDataLakeOrganizationConfiguration securitylake:CreateSubscriber securitylake:CreateSubscriberNotification securitylake:DeleteAwsLogSource securitylake:DeleteCustomLogSource securitylake:DeleteDataLakeExceptionSubscription securitylake:DeleteDataLakeOrganizationConfiguration securitylake:DeleteSubscriber securitylake:DeleteSubscriberNotification securitylake:DeregisterDataLakeDelegatedAdministrator securitylake:GetDataLakeExceptionSubscription securitylake:GetDataLakeOrganizationConfiguration securitylake:GetDataLakeSources securitylake:GetSubscriber securitylake:ListDataLakes securitylake:ListLogSources securitylake:ListSubscribers securitylake:RegisterDataLakeDelegatedAdministrator

Prefixo do serviço	Ações do
	securitylake:UpdateDataLakeExceptionSubscription securitylake:UpdateSubscriber securitylake:UpdateSubscriberNotification
serverlessrepo	serverlessrepo:CreateApplication serverlessrepo:CreateApplicationVersion serverlessrepo:CreateCloudFormationChangeSet serverlessrepo:CreateCloudFormationTemplate serverlessrepo:DeleteApplication serverlessrepo:GetApplication serverlessrepo:GetApplicationPolicy serverlessrepo:GetCloudFormationTemplate serverlessrepo:ListApplicationDependencies serverlessrepo:ListApplicationVersions serverlessrepo:ListApplications serverlessrepo:PutApplicationPolicy serverlessrepo:UnshareApplication serverlessrepo:UpdateApplication

Prefixo do serviço	Ações do
servicecatalog	servicecatalog:AcceptPortfolioShare servicecatalog:AssociateBudgetWithResource servicecatalog:AssociatePrincipalWithPortfolio servicecatalog:AssociateProductWithPortfolio servicecatalog:AssociateServiceActionWithProvisioningArtifact servicecatalog:BatchAssociateServiceActionWithProvisioningArtifact servicecatalog:BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog:CopyProduct servicecatalog:CreateAttributeGroup servicecatalog:CreateConstraint servicecatalog:CreatePortfolio servicecatalog:CreatePortfolioShare servicecatalog:CreateProduct servicecatalog:CreateProvisionedProductPlan servicecatalog:CreateProvisioningArtifact servicecatalog:CreateServiceAction servicecatalog:DeleteAttributeGroup servicecatalog:DeleteConstraint servicecatalog:DeletePortfolio servicecatalog:DeletePortfolioShare

Prefixo do serviço	Ações do
	servicecatalog:DeleteProduct servicecatalog:DeleteProvisionedProductPlan servicecatalog:DeleteProvisioningArtifact servicecatalog:DeleteServiceAction servicecatalog:DescribeConstraint servicecatalog:DescribeCopyProductStatus servicecatalog:DescribePortfolio servicecatalog:DescribePortfolioShareStatus servicecatalog:DescribePortfolioShares servicecatalog:DescribeProduct servicecatalog:DescribeProductAsAdmin servicecatalog:DescribeProductView servicecatalog:DescribeProvisionedProduct servicecatalog:DescribeProvisionedProductPlan servicecatalog:DescribeProvisioningArtifact servicecatalog:DescribeProvisioningParameters servicecatalog:DescribeRecord servicecatalog:DescribeServiceAction servicecatalog:DescribeServiceActionExecutionParameters servicecatalog:DisableAWSOrganizationsAccess servicecatalog:DisassociateBudgetFromResource

Prefixo do serviço	Ações do
	servicecatalog:DisassociatePrincipalFromPortfolio servicecatalog:DisassociateProductFromPortfolio servicecatalog:DisassociateServiceActionFromProvisioningArtifact servicecatalog:EnableAWSOrganizationsAccess servicecatalog:ExecuteProvisionedProductPlan servicecatalog:ExecuteProvisionedProductServiceAction servicecatalog:GetAWSOrganizationsAccessStatus servicecatalog:GetProvisionedProductOutputs servicecatalog:ImportAsProvisionedProduct servicecatalog:ListAcceptedPortfolioShares servicecatalog:ListAttributeGroups servicecatalog:ListBudgetsForResource servicecatalog:ListConstraintsForPortfolio servicecatalog:ListLaunchPaths servicecatalog:ListOrganizationPortfolioAccess servicecatalog:ListPortfolioAccess servicecatalog:ListPortfolios servicecatalog:ListPortfoliosForProduct servicecatalog:ListPrincipalsForPortfolio servicecatalog:ListProvisionedProductPlans servicecatalog:ListProvisioningArtifacts

Prefixo do serviço	Ações do
	servicecatalog:ListProvisioningArtifactsForServiceAction servicecatalog:ListRecordHistory servicecatalog:ListServiceActions servicecatalog:ListServiceActionsForProvisioningArtifact servicecatalog:ListStackInstancesForProvisionedProduct servicecatalog:NotifyProvisionProductEngineWorkflowResult servicecatalog:NotifyTerminateProvisionedProductEngine WorkflowResult servicecatalog:NotifyUpdateProvisionedProductEngineWor kflowResult servicecatalog:ProvisionProduct servicecatalog:RejectPortfolioShare servicecatalog:ScanProvisionedProducts servicecatalog:SearchProducts servicecatalog:SearchProductsAsAdmin servicecatalog:SearchProvisionedProducts servicecatalog:TerminateProvisionedProduct servicecatalog:UpdateConstraint servicecatalog:UpdatePortfolio servicecatalog:UpdatePortfolioShare servicecatalog:UpdateProduct servicecatalog:UpdateProvisionedProduct

Prefixo do serviço	Ações do
	servicecatalog:UpdateProvisionedProductProperties servicecatalog:UpdateProvisioningArtifact servicecatalog:UpdateServiceAction

Prefixo do serviço	Ações do
servicediscovery	servicediscovery:CreateHttpNamespace
	servicediscovery:CreatePrivateDnsNamespace
	servicediscovery:CreatePublicDnsNamespace
	servicediscovery:CreateService
	servicediscovery:DeleteNamespace
	servicediscovery:DeleteService
	servicediscovery:DeleteServiceAttributes
	servicediscovery:DeregisterInstance
	servicediscovery:GetInstance
	servicediscovery:GetInstancesHealthStatus
	servicediscovery:GetNamespace
	servicediscovery:GetOperation
	servicediscovery:GetService
	servicediscovery:ListInstances
	servicediscovery:ListNamespaces
	servicediscovery:ListOperations
	servicediscovery:ListServices
	servicediscovery:RegisterInstance
	servicediscovery:UpdateHttpNamespace
	servicediscovery:UpdateInstanceCustomHealthStatus
	servicediscovery:UpdatePrivateDnsNamespace

Prefixo do serviço	Ações do
	servicediscovery:UpdatePublicDnsNamespace servicediscovery:UpdateService servicediscovery:UpdateServiceAttributes

Prefixo do serviço	Ações do
servicequotas	servicequotas:AssociateServiceQuotaTemplate servicequotas:CreateSupportCase servicequotas:DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas:DisassociateServiceQuotaTemplate servicequotas:GetAWSDefaultServiceQuota servicequotas:GetAssociationForServiceQuotaTemplate servicequotas:GetAutoManagementConfiguration servicequotas:GetQuotaUtilizationReport servicequotas:GetRequestedServiceQuotaChange servicequotas:GetServiceQuota servicequotas:GetServiceQuotaIncreaseRequestFromTemplate servicequotas:ListAWSDefaultServiceQuotas servicequotas:ListRequestedServiceQuotaChangeHistory servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota servicequotas:ListServiceQuotaIncreaseRequestsInTemplate servicequotas:ListServiceQuotas servicequotas:ListServices servicequotas:PutServiceQuotaIncreaseRequestIntoTemplate servicequotas:RequestServiceQuotaIncrease servicequotas:StartAutoManagement servicequotas:StartQuotaUtilizationReport

Prefixo do serviço	Ações do
	servicequotas:StopAutoManagement servicequotas:UpdateAutoManagement

Prefixo do serviço	Ações do
ses	ses:BatchGetMetricData
	ses:CloneReceiptRuleSet
	ses:CreateAddonInstance
	ses:CreateAddonSubscription
	ses:CreateAddressList
	ses:CreateAddressListImportJob
	ses:CreateArchive
	ses:CreateConfigurationSet
	ses:CreateConfigurationSetEventDestination
	ses:CreateConfigurationSetTrackingOptions
	ses:CreateContact
	ses:CreateContactList
	ses:CreateCustomVerificationEmailTemplate
	ses:CreateDedicatedIpPool
	ses:CreateDeliverabilityTestReport
	ses:CreateEmailIdentity
	ses:CreateEmailIdentityPolicy
	ses:CreateEmailTemplate
	ses:CreateImportJob
	ses:CreateIngressPoint
	ses:CreateMultiRegionEndpoint

Prefixo do serviço	Ações do
	ses:CreateReceiptFilter
	ses:CreateReceiptRule
	ses:CreateReceiptRuleSet
	ses:CreateRelay
	ses:CreateRuleSet
	ses:CreateTemplate
	ses:CreateTenant
	ses:CreateTenantResourceAssociation
	ses:CreateTrafficPolicy
	ses:DeleteAddonInstance
	ses:DeleteAddonSubscription
	ses:DeleteAddressList
	ses:DeleteArchive
	ses:DeleteConfigurationSet
	ses:DeleteConfigurationSetEventDestination
	ses:DeleteConfigurationSetTrackingOptions
	ses:DeleteContact
	ses:DeleteContactList
	ses:DeleteCustomVerificationEmailTemplate
	ses:DeleteDedicatedIpPool
	ses:DeleteEmailIdentity

Prefixo do serviço	Ações do
	ses:DeleteEmailIdentityPolicy
	ses:DeleteEmailTemplate
	ses:DeleteIdentity
	ses:DeleteIdentityPolicy
	ses:DeleteIngressPoint
	ses:DeleteMultiRegionEndpoint
	ses:DeleteReceiptFilter
	ses:DeleteReceiptRule
	ses:DeleteReceiptRuleSet
	ses:DeleteRelay
	ses:DeleteRuleSet
	ses:DeleteSuppressedDestination
	ses:DeleteTemplate
	ses:DeleteTenant
	ses:DeleteTenantResourceAssociation
	ses:DeleteTrafficPolicy
	ses:DeleteVerifiedEmailAddress
	ses:DeregisterMemberFromAddressList
	ses:DescribeActiveReceiptRuleSet
	ses:DescribeConfigurationSet
	ses:DescribeReceiptRule

Prefixo do serviço	Ações do
	ses:DescribeReceiptRuleSet
	ses:GetAccount
	ses:GetAccountSendingEnabled
	ses:GetAddonInstance
	ses:GetAddonSubscription
	ses:GetAddressList
	ses:GetArchive
	ses:GetArchiveExport
	ses:GetArchiveMessage
	ses:GetArchiveMessageContent
	ses:GetArchiveSearch
	ses:GetArchiveSearchResults
	ses:GetBlacklistReports
	ses:GetConfigurationSet
	ses:GetConfigurationSetEventDestinations
	ses:GetContact
	ses:GetContactList
	ses:GetCustomVerificationEmailTemplate
	ses:GetDedicatedIp
	ses:GetDedicatedIpPool
	ses:GetDedicatedIps

Prefixo do serviço	Ações do
	ses:GetDeliverabilityDashboardOptions
	ses:GetDeliverabilityTestReport
	ses:GetDomainDeliverabilityCampaign
	ses:GetDomainStatisticsReport
	ses:GetEmailAddressInsights
	ses:GetEmailIdentity
	ses:GetEmailIdentityPolicies
	ses:GetEmailTemplate
	ses:GetIdentityDkimAttributes
	ses:GetIdentityMailFromDomainAttributes
	ses:GetIdentityNotificationAttributes
	ses:GetIdentityPolicies
	ses:GetIdentityVerificationAttributes
	ses:GetImportJob
	ses:GetIngressPoint
	ses:GetMemberOfAddressList
	ses:GetMessageInsights
	ses:GetMultiRegionEndpoint
	ses:GetRelay
	ses:GetRuleSet
	ses:GetSendQuota

Prefixo do serviço	Ações do
	ses:GetSendStatistics
	ses:GetSuppressedDestination
	ses:GetTemplate
	ses:GetTenant
	ses:GetTrafficPolicy
	ses:ListAddonInstances
	ses:ListAddonSubscriptions
	ses:ListAddressListImportJobs
	ses:ListAddressLists
	ses:ListArchiveExports
	ses:ListArchiveSearches
	ses:ListArchives
	ses:ListConfigurationSets
	ses:ListContactLists
	ses:ListContacts
	ses:ListCustomVerificationEmailTemplates
	ses:ListDedicatedIpPools
	ses:ListDeliverabilityTestReports
	ses:ListDomainDeliverabilityCampaigns
	ses:ListEmailIdentities
	ses:ListEmailTemplates

Prefixo do serviço	Ações do
	ses:ListExportJobs
	ses:ListIdentities
	ses:ListIdentityPolicies
	ses:ListImportJobs
	ses:ListIngressPoints
	ses:ListMembersOfAddressList
	ses:ListMultiRegionEndpoints
	ses:ListReceiptFilters
	ses:ListReceiptRuleSets
	ses:ListRecommendations
	ses:ListRelays
	ses:ListReputationEntities
	ses:ListResourceTenants
	ses:ListRuleSets
	ses:ListSuppressedDestinations
	ses:ListTemplates
	ses:ListTenantResources
	ses:ListTenants
	ses:ListTrafficPolicies
	ses:ListVerifiedEmailAddresses
	ses:PutAccountDedicatedIpWarmupAttributes

Prefixo do serviço	Ações do
	ses:PutAccountDetails
	ses:PutAccountSendingAttributes
	ses:PutAccountSuppressionAttributes
	ses:PutAccountVdmAttributes
	ses:PutConfigurationSetArchivingOptions
	ses:PutConfigurationSetDeliveryOptions
	ses:PutConfigurationSetReputationOptions
	ses:PutConfigurationSetSendingOptions
	ses:PutConfigurationSetSuppressionOptions
	ses:PutConfigurationSetTrackingOptions
	ses:PutConfigurationSetVdmOptions
	ses:PutDedicatedIpInPool
	ses:PutDedicatedIpPoolScalingAttributes
	ses:PutDedicatedIpWarmupAttributes
	ses:PutDeliverabilityDashboardOption
	ses:PutEmailIdentityConfigurationSetAttributes
	ses:PutEmailIdentityDkimAttributes
	ses:PutEmailIdentityDkimSigningAttributes
	ses:PutEmailIdentityFeedbackAttributes
	ses:PutEmailIdentityMailFromAttributes
	ses:PutIdentityPolicy

Prefixo do serviço	Ações do
	ses:PutSuppressedDestination
	ses:PutTenantSuppressionAttributes
	ses:RegisterMemberToAddressList
	ses:ReorderReceiptRuleSet
	ses:SendBounce
	ses:SendCustomVerificationEmail
	ses:SetActiveReceiptRuleSet
	ses:SetIdentityDkimEnabled
	ses:SetIdentityFeedbackForwardingEnabled
	ses:SetIdentityHeadersInNotificationsEnabled
	ses:SetIdentityMailFromDomain
	ses:SetIdentityNotificationTopic
	ses:SetReceiptRulePosition
	ses:StartArchiveExport
	ses:StartArchiveSearch
	ses:StopArchiveExport
	ses:StopArchiveSearch
	ses:TestRenderEmailTemplate
	ses:TestRenderTemplate
	ses:UpdateAccountSendingEnabled
	ses:UpdateArchive

Prefixo do serviço	Ações do
	ses:UpdateConfigurationSetEventDestination
	ses:UpdateConfigurationSetReputationMetricsEnabled
	ses:UpdateConfigurationSetSendingEnabled
	ses:UpdateConfigurationSetTrackingOptions
	ses:UpdateContact
	ses:UpdateContactList
	ses:UpdateCustomVerificationEmailTemplate
	ses:UpdateEmailIdentityPolicy
	ses:UpdateEmailTemplate
	ses:UpdateIngressPoint
	ses:UpdateReceiptRule
	ses:UpdateRelay
	ses:UpdateRuleSet
	ses:UpdateTemplate
	ses:UpdateTrafficPolicy
	ses:VerifyDomainDkim
	ses:VerifyDomainIdentity
	ses:VerifyEmailAddress
	ses:VerifyEmailIdentity

Prefixo do serviço	Ações do
shield	shield:AssociateDRTLogBucket shield:AssociateHealthCheck shield:AssociateProactiveEngagementDetails shield:CreateProtection shield:CreateProtectionGroup shield:CreateSubscription shield:DeleteProtection shield:DeleteProtectionGroup shield:DeleteSubscription shield:DescribeAttack shield:DescribeAttackStatistics shield:DescribeDRTAccess shield:DescribeEmergencyContactSettings shield:DescribeProtection shield:DescribeProtectionGroup shield:DescribeSubscription shield:DisableApplicationLayerAutomaticResponse shield:DisableProactiveEngagement shield:DisassociateDRTLogBucket shield:DisassociateDRTRole shield:DisassociateHealthCheck

Prefixo do serviço	Ações do
	shield:EnableApplicationLayerAutomaticResponse shield:EnableProactiveEngagement shield:GetSubscriptionState shield:ListAttacks shield:ListProtectionGroups shield:ListProtections shield:ListResourcesInProtectionGroup shield:UpdateApplicationLayerAutomaticResponse shield:UpdateEmergencyContactSettings shield:UpdateProtectionGroup shield:UpdateSubscription

Prefixo do serviço	Ações do
signer	signer:AddProfilePermission
	signer:CancelSigningProfile
	signer:DescribeSigningJob
	signer:GetSigningPlatform
	signer:GetSigningProfile
	signer:ListProfilePermissions
	signer:ListSigningJobs
	signer:ListSigningPlatforms
	signer:ListSigningProfiles
	signer:PutSigningProfile
	signer:RemoveProfilePermission
	signer:RevokeSignature
	signer:RevokeSigningProfile
	signer:SignPayload
	signer:StartSigningJob

Prefixo do serviço	Ações do
simspaceweaver	simspaceweaver:CreateSnapshot simspaceweaver:DeleteApp simspaceweaver:DeleteSimulation simspaceweaver:DescribeApp simspaceweaver:DescribeSimulation simspaceweaver:ListApps simspaceweaver:ListSimulations simspaceweaver:StartApp simspaceweaver:StartClock simspaceweaver:StartSimulation simspaceweaver:StopApp simspaceweaver:StopClock simspaceweaver:StopSimulation

Prefixo do serviço	Ações do
sms	sms:CreateApp
	sms:CreateReplicationJob
	sms:DeleteApp
	sms:DeleteAppLaunchConfiguration
	sms:DeleteAppReplicationConfiguration
	sms:DeleteAppValidationConfiguration
	sms:DeleteReplicationJob
	sms:DeleteServerCatalog
	sms:DisassociateConnector
	sms:GenerateChangeSet
	sms:GenerateTemplate
	sms:GetApp
	sms:GetAppLaunchConfiguration
	sms:GetAppReplicationConfiguration
	sms:GetAppValidationConfiguration
	sms:GetAppValidationOutput
	sms:GetConnectors
	sms:GetReplicationJobs
	sms:GetReplicationRuns
	sms:GetServers
	sms:ImportAppCatalog

Prefixo do serviço	Ações do
	sms:ImportServerCatalog
	sms:LaunchApp
	sms:ListApps
	sms:NotifyAppValidationOutput
	sms:PutAppLaunchConfiguration
	sms:PutAppReplicationConfiguration
	sms:PutAppValidationConfiguration
	sms:StartAppReplication
	sms:StartOnDemandAppReplication
	sms:StartOnDemandReplicationRun
	sms:StopAppReplication
	sms:TerminateApp
	sms:UpdateApp
	sms:UpdateReplicationJob

Prefixo do serviço	Ações do
sms-voice	sms-voice:AssociateProtectConfiguration sms-voice:CreateConfigurationSet sms-voice:CreateConfigurationSetEventDestination sms-voice:CreateEventDestination sms-voice:CreateOptOutList sms-voice:CreatePool sms-voice:CreateProtectConfiguration sms-voice:CreateRegistration sms-voice:CreateRegistrationAssociation sms-voice:CreateRegistrationAttachment sms-voice:CreateRegistrationVersion sms-voice:CreateVerifiedDestinationNumber sms-voice:DeleteAccountDefaultProtectConfiguration sms-voice:DeleteConfigurationSet sms-voice:DeleteConfigurationSetEventDestination sms-voice:DeleteDefaultMessageType sms-voice:DeleteDefaultSenderId sms-voice:DeleteEventDestination sms-voice:DeleteKeyword sms-voice:DeleteMediaMessageSpendLimitOverride sms-voice:DeleteOptOutList

Prefixo do serviço	Ações do
	sms-voice:DeleteOptedOutNumber sms-voice:DeletePool sms-voice:DeleteProtectConfiguration sms-voice:DeleteProtectConfigurationRuleSetNumberOverride sms-voice:DeleteRegistration sms-voice:DeleteRegistrationAttachment sms-voice:DeleteResourcePolicy sms-voice:DeleteTextMessageSpendLimitOverride sms-voice:DeleteVerifiedDestinationNumber sms-voice:DeleteVoiceMessageSpendLimitOverride sms-voice:DescribeAccountAttributes sms-voice:DescribeAccountLimits sms-voice:DescribeConfigurationSets sms-voice:DescribeKeywords sms-voice:DescribeOptOutLists sms-voice:DescribeOptedOutNumbers sms-voice:DescribePhoneNumbers sms-voice:DescribePools sms-voice:DescribeProtectConfigurations sms-voice:DescribeRegistrationAttachments sms-voice:DescribeRegistrationFieldDefinitions

Prefixo do serviço	Ações do
	sms-voice:DescribeRegistrationFieldValues sms-voice:DescribeRegistrationSectionDefinitions sms-voice:DescribeRegistrationTypeDefinitions sms-voice:DescribeRegistrationVersions sms-voice:DescribeRegistrations sms-voice:DescribeSenderIds sms-voice:DescribeSpendLimits sms-voice:DescribeVerifiedDestinationNumbers sms-voice:DisassociateOriginationIdentity sms-voice:DisassociateProtectConfiguration sms-voice:DiscardRegistrationVersion sms-voice:GetConfigurationSetEventDestinations sms-voice:GetProtectConfigurationCountryRuleSet sms-voice:GetResourcePolicy sms-voice:ListConfigurationSets sms-voice:ListPoolOriginationIdentities sms-voice:ListProtectConfigurationRuleSetNumberOverrides sms-voice:ListRegistrationAssociations sms-voice:PutKeyword sms-voice:PutOptedOutNumber sms-voice:PutProtectConfigurationRuleSetNumberOverride

Prefixo do serviço	Ações do
	sms-voice:PutResourcePolicy sms-voice:ReleasePhoneNumber sms-voice:ReleaseSenderId sms-voice:RequestPhoneNumber sms-voice:RequestSenderId sms-voice:SendDestinationNumberVerificationCode sms-voice:SetAccountDefaultProtectConfiguration sms-voice:SetDefaultMessageFeedbackEnabled sms-voice:SetDefaultMessageType sms-voice:SetDefaultSenderId sms-voice:SetMediaMessageSpendLimitOverride sms-voice:SetTextMessageSpendLimitOverride sms-voice:SetVoiceMessageSpendLimitOverride sms-voice:SubmitRegistrationVersion sms-voice:UpdateConfigurationSetEventDestination sms-voice:UpdateEventDestination sms-voice:UpdatePhoneNumber sms-voice:UpdatePool sms-voice:UpdateProtectConfiguration sms-voice:UpdateProtectConfigurationCountryRuleSet sms-voice:UpdateSenderId

Prefixo do serviço	Ações do
snowball	snowball:CancelCluster
	snowball:CancelJob
	snowball:CreateAddress
	snowball:CreateCluster
	snowball:CreateJob
	snowball:CreateLongTermPricing
	snowball:CreateReturnShippingLabel
	snowball:DescribeAddress
	snowball:DescribeAddresses
	snowball:DescribeCluster
	snowball:DescribeJob
	snowball:DescribeReturnShippingLabel
	snowball:GetJobManifest
	snowball:GetJobUnlockCode
	snowball:GetSnowballUsage
	snowball:GetSoftwareUpdates
	snowball:ListClusterJobs
	snowball:ListClusters
	snowball:ListCompatibleImages
	snowball:ListJobs
	snowball:ListLongTermPricing

Prefixo do serviço	Ações do
	snowball:ListPickupLocations snowball:ListServiceVersions snowball:UpdateCluster snowball:UpdateJob snowball:UpdateJobShipmentState snowball:UpdateLongTermPricing
sqs	sqs:AddPermission sqs:CancelMessageMoveTask sqs:CreateQueue sqs:DeleteQueue sqs:PurgeQueue sqs:RemovePermission sqs:SetQueueAttributes

Prefixo do serviço	Ações do
ssm	ssm:AssociateOpsItemRelatedItem
	ssm:CancelCommand
	ssm:CancelMaintenanceWindowExecution
	ssm:CreateActivation
	ssm:CreateAssociation
	ssm:CreateAssociationBatch
	ssm:CreateDocument
	ssm:CreateMaintenanceWindow
	ssm:CreateOpsItem
	ssm:CreateOpsMetadata
	ssm:CreatePatchBaseline
	ssm:CreateResourceDataSync
	ssm:DeleteActivation
	ssm:DeleteAssociation
	ssm:DeleteCloudConnector
	ssm:DeleteDocument
	ssm:DeleteInventory
	ssm:DeleteMaintenanceWindow
	ssm:DeleteOpsItem
	ssm:DeleteOpsMetadata
	ssm:DeleteParameter

Prefixo do serviço	Ações do
	ssm:DeleteParameters
	ssm:DeletePatchBaseline
	ssm:DeleteResourceDataSync
	ssm:DeleteResourcePolicy
	ssm:DeregisterManagedInstance
	ssm:DeregisterPatchBaselineForPatchGroup
	ssm:DeregisterTargetFromMaintenanceWindow
	ssm:DeregisterTaskFromMaintenanceWindow
	ssm:DescribeActivations
	ssm:DescribeAssociation
	ssm:DescribeAssociationExecutionTargets
	ssm:DescribeAssociationExecutions
	ssm:DescribeAutomationExecutions
	ssm:DescribeAutomationStepExecutions
	ssm:DescribeAvailablePatches
	ssm:DescribeDocument
	ssm:DescribeDocumentParameters
	ssm:DescribeDocumentPermission
	ssm:DescribeEffectiveInstanceAssociations
	ssm:DescribeEffectivePatchesForPatchBaseline
	ssm:DescribeInstanceAssociationsStatus

Prefixo do serviço	Ações do
	ssm:DescribeInstanceInformation
	ssm:DescribeInstancePatchStates
	ssm:DescribeInstancePatchStatesForPatchGroup
	ssm:DescribeInstancePatches
	ssm:DescribeInstanceProperties
	ssm:DescribeInventoryDeletions
	ssm:DescribeMaintenanceWindowExecutionTaskInvocations
	ssm:DescribeMaintenanceWindowExecutionTasks
	ssm:DescribeMaintenanceWindowExecutions
	ssm:DescribeMaintenanceWindowSchedule
	ssm:DescribeMaintenanceWindowTargets
	ssm:DescribeMaintenanceWindowTasks
	ssm:DescribeMaintenanceWindows
	ssm:DescribeMaintenanceWindowsForTarget
	ssm:DescribeOpsItems
	ssm:DescribeParameters
	ssm:DescribePatchBaselines
	ssm:DescribePatchGroupState
	ssm:DescribePatchGroups
	ssm:DescribePatchProperties
	ssm:DescribeSessions

Prefixo do serviço	Ações do
	ssm:DisassociateOpsItemRelatedItem
	ssm:GetAccessToken
	ssm:GetAutomationExecution
	ssm:GetCalendarState
	ssm:GetCloudConnector
	ssm:GetCommandInvocation
	ssm:GetConnectionStatus
	ssm:GetDefaultPatchBaseline
	ssm:GetDeployablePatchSnapshotForInstance
	ssm:GetDocument
	ssm:GetExecutionPreview
	ssm:GetInventory
	ssm:GetInventorySchema
	ssm:GetMaintenanceWindow
	ssm:GetMaintenanceWindowExecution
	ssm:GetMaintenanceWindowExecutionTask
	ssm:GetMaintenanceWindowExecutionTaskInvocation
	ssm:GetMaintenanceWindowTask
	ssm:GetOpsItem
	ssm:GetOpsMetadata
	ssm:GetOpsSummary

Prefixo do serviço	Ações do
	ssm:GetParameter
	ssm:GetParameterHistory
	ssm:GetParameters
	ssm:GetParametersByPath
	ssm:GetPatchBaseline
	ssm:GetPatchBaselineForPatchGroup
	ssm:GetResourcePolicies
	ssm:GetServiceSetting
	ssm:LabelParameterVersion
	ssm:ListAssociationVersions
	ssm:ListAssociations
	ssm:ListCloudConnectors
	ssm:ListCommandInvocations
	ssm:ListCommands
	ssm:ListComplianceItems
	ssm:ListComplianceSummaries
	ssm:ListDocumentMetadataHistory
	ssm:ListDocumentVersions
	ssm:ListDocuments
	ssm:ListInstanceAssociations
	ssm:ListInventoryEntries

Prefixo do serviço	Ações do
	ssm:ListNodes
	ssm:ListNodesSummary
	ssm:ListOpsItemEvents
	ssm:ListOpsItemRelatedItems
	ssm:ListOpsMetadata
	ssm:ListResourceComplianceSummaries
	ssm:ListResourceDataSync
	ssm:ModifyDocumentPermission
	ssm:PutComplianceItems
	ssm:PutInventory
	ssm:PutParameter
	ssm:PutResourcePolicy
	ssm:RegisterDefaultPatchBaseline
	ssm:RegisterManagedInstance
	ssm:RegisterPatchBaselineForPatchGroup
	ssm:RegisterTargetWithMaintenanceWindow
	ssm:RegisterTaskWithMaintenanceWindow
	ssm:ResetServiceSetting
	ssm:ResumeSession
	ssm:SendAutomationSignal
	ssm:SendCommand

Prefixo do serviço	Ações do
	ssm:StartAssociationsOnce
	ssm:StartAutomationExecution
	ssm:StartChangeRequestExecution
	ssm:StartSession
	ssm:StopAutomationExecution
	ssm:TerminateSession
	ssm:UnlabelParameterVersion
	ssm:UpdateAssociation
	ssm:UpdateAssociationStatus
	ssm:UpdateCloudConnector
	ssm:UpdateDocument
	ssm:UpdateDocumentDefaultVersion
	ssm:UpdateDocumentMetadata
	ssm:UpdateInstanceInformation
	ssm:UpdateMaintenanceWindow
	ssm:UpdateMaintenanceWindowTarget
	ssm:UpdateMaintenanceWindowTask
	ssm:UpdateManagedInstanceRole
	ssm:UpdateOpsItem
	ssm:UpdateOpsMetadata
	ssm:UpdatePatchBaseline

Prefixo do serviço	Ações do
	ssm:UpdateResourceDataSync ssm:UpdateServiceSetting ssm:ValidateCloudConnector

Prefixo do serviço	Ações do
ssm-incidents	ssm-incidents:BatchGetIncidentFindings ssm-incidents:CreateReplicationSet ssm-incidents:CreateResponsePlan ssm-incidents:CreateTimelineEvent ssm-incidents:DeleteIncidentRecord ssm-incidents:DeleteReplicationSet ssm-incidents:DeleteResourcePolicy ssm-incidents:DeleteResponsePlan ssm-incidents:DeleteTimelineEvent ssm-incidents:GetIncidentRecord ssm-incidents:GetReplicationSet ssm-incidents:GetResourcePolicies ssm-incidents:GetResponsePlan ssm-incidents:GetTimelineEvent ssm-incidents:ListIncidentFindings ssm-incidents:ListIncidentRecords ssm-incidents:ListRelatedItems ssm-incidents:ListReplicationSets ssm-incidents:ListResponsePlans ssm-incidents:ListTimelineEvents ssm-incidents:PutResourcePolicy

Prefixo do serviço	Ações do
	ssm-incidents:StartIncident ssm-incidents:UpdateDeletionProtection ssm-incidents:UpdateIncidentRecord ssm-incidents:UpdateRelatedItems ssm-incidents:UpdateReplicationSet ssm-incidents:UpdateResponsePlan ssm-incidents:UpdateTimelineEvent

Prefixo do serviço	Ações do
ssm-sap	ssm-sap:BackupDatabase
	ssm-sap:DeleteResourcePermission
	ssm-sap:DeregisterApplication
	ssm-sap:GetApplication
	ssm-sap:GetComponent
	ssm-sap:GetConfigurationCheckOperation
	ssm-sap:GetDatabase
	ssm-sap:GetOperation
	ssm-sap:GetResourcePermission
	ssm-sap:ListApplications
	ssm-sap:ListComponents
	ssm-sap:ListConfigurationCheckDefinitions
	ssm-sap:ListConfigurationCheckOperations
	ssm-sap:ListDatabases
	ssm-sap:ListOperationEvents
	ssm-sap:ListOperations
	ssm-sap:ListSubCheckResults
	ssm-sap:ListSubCheckRuleResults
	ssm-sap:PutResourcePermission
	ssm-sap:RegisterApplication
	ssm-sap:RestoreDatabase

Prefixo do serviço	Ações do
	ssm-sap:StartApplication
	ssm-sap:StartApplicationRefresh
	ssm-sap:StartConfigurationChecks
	ssm-sap:StopApplication
	ssm-sap:UpdateApplicationSettings
	ssm-sap:UpdateHANABackupSettings

Prefixo do serviço	Ações do
estados	states:CreateActivity states:CreateStateMachine states:CreateStateMachineAlias states:DeleteActivity states:DeleteStateMachine states:DeleteStateMachineAlias states:DeleteStateMachineVersion states:DescribeActivity states:DescribeExecution states:DescribeMapRun states:DescribeStateMachine states:DescribeStateMachineAlias states:DescribeStateMachineForExecution states:GetExecutionHistory states:ListActivities states:ListExecutions states:ListMapRuns states:ListStateMachineAliases states:ListStateMachineVersions states:ListStateMachines states:SendTaskFailure

Prefixo do serviço	Ações do
	states:SendTaskHeartbeat states:SendTaskSuccess states:StartExecution states:StopExecution states:UpdateMapRun states:UpdateStateMachine states:UpdateStateMachineAlias states:ValidateStateMachineDefinition
sts	sts:AssumeRole sts:AssumeRoleWithSAML sts:AssumeRoleWithWebIdentity sts:DecodeAuthorizationMessage sts:GetAccessKeyInfo sts:GetCallerIdentity sts:GetFederationToken sts:GetSessionToken sts:GetWebIdentityToken

Prefixo do serviço	Ações do
swf	swf:DeleteActivityType swf:DeleteWorkflowType swf:DeprecateActivityType swf:DeprecateDomain swf:DeprecateWorkflowType swf:DescribeActivityType swf:DescribeDomain swf:DescribeWorkflowType swf:ListActivityTypes swf:ListDomains swf:ListWorkflowTypes swf:RegisterActivityType swf:RegisterDomain swf:RegisterWorkflowType swf:UndeprecateActivityType swf:UndeprecateDomain swf:UndeprecateWorkflowType

Prefixo do serviço	Ações do
synthetics	synthetics:AssociateResource
	synthetics:CreateCanary
	synthetics:CreateGroup
	synthetics:DeleteCanary
	synthetics:DeleteGroup
	synthetics:DescribeCanaries
	synthetics:DescribeCanariesLastRun
	synthetics:DescribeRuntimeVersions
	synthetics:DisassociateResource
	synthetics:GetCanary
	synthetics:GetCanaryRuns
	synthetics:GetGroup
	synthetics:ListAssociatedGroups
	synthetics:ListGroupResources
	synthetics:ListGroups
	synthetics:StartCanary
	synthetics:StartCanaryDryRun
	synthetics:StopCanary
	synthetics:UpdateCanary

Prefixo do serviço	Ações do
tag	tag:DescribeReportCreation tag:GetComplianceSummary tag:GetResources tag:StartReportCreation

Prefixo do serviço	Ações do
textract	textract:AnalyzeDocument
	textract:AnalyzeExpense
	textract:AnalyzeID
	textract:CreateAdapter
	textract:CreateAdapterVersion
	textract:DeleteAdapter
	textract:DeleteAdapterVersion
	textract:DetectDocumentText
	textract:GetAdapter
	textract:GetAdapterVersion
	textract:GetDocumentAnalysis
	textract:GetDocumentTextDetection
	textract:GetExpenseAnalysis
	textract:GetLendingAnalysis
	textract:GetLendingAnalysisSummary
	textract:ListAdapterVersions
	textract:ListAdapters
	textract:StartDocumentAnalysis
	textract:StartDocumentTextDetection
	textract:StartExpenseAnalysis
	textract:StartLendingAnalysis

Prefixo do serviço	Ações do
	textract:UpdateAdapter

Prefixo do serviço	Ações do
timestream	timestream:CancelQuery timestream:CreateDatabase timestream:CreateScheduledQuery timestream:CreateTable timestream:DeleteDatabase timestream:DeleteScheduledQuery timestream:DeleteTable timestream:DescribeAccountSettings timestream:DescribeDatabase timestream:DescribeScheduledQuery timestream:DescribeTable timestream:ExecuteScheduledQuery timestream:ListBatchLoadTasks timestream:ListDatabases timestream:ListScheduledQueries timestream:ListTables timestream:PrepareQuery timestream:UpdateAccountSettings timestream:UpdateDatabase timestream:UpdateScheduledQuery timestream:UpdateTable

Prefixo do serviço	Ações do
tnb	tnb:CancelSolNetworkOperation
	tnb:CreateSolFunctionPackage
	tnb:CreateSolNetworkInstance
	tnb:CreateSolNetworkPackage
	tnb:DeleteSolFunctionPackage
	tnb:DeleteSolNetworkInstance
	tnb:DeleteSolNetworkPackage
	tnb:GetSolFunctionInstance
	tnb:GetSolFunctionPackage
	tnb:GetSolFunctionPackageContent
	tnb:GetSolFunctionPackageDescriptor
	tnb:GetSolNetworkInstance
	tnb:GetSolNetworkOperation
	tnb:GetSolNetworkPackage
	tnb:GetSolNetworkPackageContent
	tnb:GetSolNetworkPackageDescriptor
	tnb:InstantiateSolNetworkInstance
	tnb:ListSolFunctionInstances
	tnb:ListSolFunctionPackages
	tnb:ListSolNetworkInstances
	tnb:ListSolNetworkOperations

Prefixo do serviço	Ações do
	tnb:ListSolNetworkPackages
	tnb:PutSolFunctionPackageContent
	tnb:PutSolNetworkPackageContent
	tnb:TerminateSolNetworkInstance
	tnb:UpdateSolFunctionPackage
	tnb:UpdateSolNetworkInstance
	tnb:UpdateSolNetworkPackage
	tnb:ValidateSolFunctionPackageContent
	tnb:ValidateSolNetworkPackageContent

Prefixo do serviço	Ações do
transcribe	transcribe:CreateCallAnalyticsCategory transcribe:CreateLanguageModel transcribe:CreateMedicalVocabulary transcribe:CreateVocabulary transcribe:CreateVocabularyFilter transcribe:DeleteCallAnalyticsCategory transcribe:DeleteCallAnalyticsJob transcribe:DeleteLanguageModel transcribe:DeleteMedicalScribeJob transcribe:DeleteMedicalTranscriptionJob transcribe:DeleteMedicalVocabulary transcribe:DeleteTranscriptionJob transcribe:DeleteVocabulary transcribe:DeleteVocabularyFilter transcribe:DescribeLanguageModel transcribe:GetCallAnalyticsCategory transcribe:GetCallAnalyticsJob transcribe:GetMedicalScribeJob transcribe:GetMedicalTranscriptionJob transcribe:GetMedicalVocabulary transcribe:GetTranscriptionJob

Prefixo do serviço	Ações do
	transcribe:GetVocabulary transcribe:GetVocabularyFilter transcribe:ListCallAnalyticsCategories transcribe:ListCallAnalyticsJobs transcribe:ListLanguageModels transcribe:ListMedicalScribeJobs transcribe:ListMedicalTranscriptionJobs transcribe:ListMedicalVocabularies transcribe:ListTranscriptionJobs transcribe:ListVocabularies transcribe:ListVocabularyFilters transcribe:StartCallAnalyticsJob transcribe:StartCallAnalyticsStreamTranscription transcribe:StartCallAnalyticsStreamTranscriptionWebSocket transcribe:StartMedicalScribeJob transcribe:StartMedicalStreamTranscription transcribe:StartMedicalStreamTranscriptionWebSocket transcribe:StartMedicalTranscriptionJob transcribe:StartStreamTranscription transcribe:StartStreamTranscriptionWebSocket transcribe:StartTranscriptionJob

Prefixo do serviço	Ações do
	transcribe:UpdateCallAnalyticsCategory transcribe:UpdateMedicalVocabulary transcribe:UpdateVocabulary transcribe:UpdateVocabularyFilter

Prefixo do serviço	Ações do
transferência	transfer:CreateAccess
	transfer:CreateAgreement
	transfer:CreateConnector
	transfer:CreateProfile
	transfer:CreateServer
	transfer:CreateUser
	transfer:CreateWebApp
	transfer:CreateWorkflow
	transfer:DeleteAccess
	transfer:DeleteAgreement
	transfer:DeleteCertificate
	transfer:DeleteConnector
	transfer:DeleteHostKey
	transfer:DeleteProfile
	transfer:DeleteServer
	transfer:DeleteSshPublicKey
	transfer:DeleteUser
	transfer:DeleteWebApp
	transfer:DeleteWebAppCustomization
	transfer:DeleteWorkflow
	transfer:DescribeAccess

Prefixo do serviço	Ações do
	transfer:DescribeAgreement
	transfer:DescribeCertificate
	transfer:DescribeConnector
	transfer:DescribeExecution
	transfer:DescribeHostKey
	transfer:DescribeProfile
	transfer:DescribeSecurityPolicy
	transfer:DescribeServer
	transfer:DescribeUser
	transfer:DescribeWebApp
	transfer:DescribeWebAppCustomization
	transfer:DescribeWorkflow
	transfer:ImportCertificate
	transfer:ImportHostKey
	transfer:ImportSshPublicKey
	transfer:ListAccesses
	transfer:ListCertificates
	transfer:ListConnectors
	transfer:ListExecutions
	transfer:ListFileTransferResults
	transfer:ListHostKeys

Prefixo do serviço	Ações do
	transfer:ListProfiles
	transfer:ListSecurityPolicies
	transfer:ListServers
	transfer:ListUsers
	transfer:ListWebApps
	transfer:ListWorkflows
	transfer:SendWorkflowStepState
	transfer:StartDirectoryListing
	transfer:StartFileTransfer
	transfer:StartRemoteDelete
	transfer:StartRemoteMove
	transfer:StartServer
	transfer:StopServer
	transfer:TestConnection
	transfer:TestIdentityProvider
	transfer:UpdateAccess
	transfer:UpdateAgreement
	transfer:UpdateCertificate
	transfer:UpdateConnector
	transfer:UpdateHostKey
	transfer:UpdateProfile

Prefixo do serviço	Ações do
	transfer:UpdateServer transfer:UpdateUser transfer:UpdateWebApp transfer:UpdateWebAppCustomization
translate	translate:CreateParallelData translate:DeleteParallelData translate:DeleteTerminology translate:DescribeTextTranslationJob translate:GetParallelData translate:GetTerminology translate:ImportTerminology translate:ListLanguages translate:ListParallelData translate:ListTerminologies translate:ListTextTranslationJobs translate:StartTextTranslationJob translate:StopTextTranslationJob translate:TranslateDocument translate:TranslateText translate:UpdateParallelData

Prefixo do serviço	Ações do
voiceid	voiceid:AssociateFraudster
	voiceid:CreateDomain
	voiceid:CreateWatchlist
	voiceid:DeleteDomain
	voiceid:DeleteFraudster
	voiceid:DeleteSpeaker
	voiceid:DeleteWatchlist
	voiceid:DescribeDomain
	voiceid:DescribeFraudster
	voiceid:DescribeFraudsterRegistrationJob
	voiceid:DescribeSpeaker
	voiceid:DescribeSpeakerEnrollmentJob
	voiceid:DescribeWatchlist
	voiceid:DisassociateFraudster
	voiceid:EvaluateSession
	voiceid:ListDomains
	voiceid:ListFraudsterRegistrationJobs
	voiceid:ListFraudsters
	voiceid:ListSpeakerEnrollmentJobs
	voiceid:ListSpeakers
	voiceid:ListWatchlists

Prefixo do serviço	Ações do
	voiceid:OptOutSpeaker
	voiceid:StartFraudsterRegistrationJob
	voiceid:StartSpeakerEnrollmentJob
	voiceid:UpdateDomain
	voiceid:UpdateWatchlist

Prefixo do serviço	Ações do
vpc-lattice	vpc-lattice:CreateAccessLogSubscription vpc-lattice:CreateListener vpc-lattice:CreateResourceConfiguration vpc-lattice:CreateResourceGateway vpc-lattice:CreateRule vpc-lattice:CreateService vpc-lattice:CreateServiceNetwork vpc-lattice:CreateServiceNetworkResourceAssociation vpc-lattice:CreateServiceNetworkServiceAssociation vpc-lattice:CreateServiceNetworkVpcAssociation vpc-lattice:CreateTargetGroup vpc-lattice:DeleteAccessLogSubscription vpc-lattice:DeleteAuthPolicy vpc-lattice:DeleteDomainVerification vpc-lattice:DeleteListener vpc-lattice:DeleteResourceConfiguration vpc-lattice:DeleteResourceEndpointAssociation vpc-lattice:DeleteResourceGateway vpc-lattice:DeleteResourcePolicy vpc-lattice:DeleteRule vpc-lattice:DeleteService

Prefixo do serviço	Ações do
	vpc-lattice:DeleteServiceNetwork vpc-lattice:DeleteServiceNetworkResourceAssociation vpc-lattice:DeleteServiceNetworkServiceAssociation vpc-lattice:DeleteServiceNetworkVpcAssociation vpc-lattice:DeleteTargetGroup vpc-lattice:DeregisterTargets vpc-lattice:GetAccessLogSubscription vpc-lattice:GetAuthPolicy vpc-lattice:GetDomainVerification vpc-lattice:GetListener vpc-lattice:GetResourceConfiguration vpc-lattice:GetResourceGateway vpc-lattice:GetResourcePolicy vpc-lattice:GetRule vpc-lattice:GetService vpc-lattice:GetServiceNetwork vpc-lattice:GetServiceNetworkResourceAssociation vpc-lattice:GetServiceNetworkServiceAssociation vpc-lattice:GetServiceNetworkVpcAssociation vpc-lattice:GetTargetGroup vpc-lattice:ListAccessLogSubscriptions

Prefixo do serviço	Ações do
	vpc-lattice:ListDomainVerifications vpc-lattice:ListListeners vpc-lattice:ListResourceConfigurations vpc-lattice:ListResourceEndpointAssociations vpc-lattice:ListResourceGateways vpc-lattice:ListRules vpc-lattice:ListServiceNetworkResourceAssociations vpc-lattice:ListServiceNetworkServiceAssociations vpc-lattice:ListServiceNetworkVpcAssociations vpc-lattice:ListServiceNetworkVpcEndpointAssociations vpc-lattice:ListServiceNetworks vpc-lattice:ListServices vpc-lattice:ListTargetGroups vpc-lattice:ListTargets vpc-lattice:PutAuthPolicy vpc-lattice:PutResourcePolicy vpc-lattice:RegisterTargets vpc-lattice:StartDomainVerification vpc-lattice:UpdateAccessLogSubscription vpc-lattice:UpdateListener vpc-lattice:UpdateResourceConfiguration

Prefixo do serviço	Ações do
	vpc-lattice:UpdateResourceGateway vpc-lattice:UpdateRule vpc-lattice:UpdateService vpc-lattice:UpdateServiceNetwork vpc-lattice:UpdateServiceNetworkVpcAssociation vpc-lattice:UpdateTargetGroup

Prefixo do serviço	Ações do
wafv2	wafv2:AssociateWebACL
	wafv2:CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2:CreateRegexPatternSet
	wafv2:CreateRuleGroup
	wafv2:CreateWebACL
	wafv2:DeleteAPIKey
	wafv2:DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2:DeleteLoggingConfiguration
	wafv2:DeletePermissionPolicy
	wafv2:DeleteRegexPatternSet
	wafv2:DeleteRuleGroup
	wafv2:DeleteWebACL
	wafv2:DescribeAllManagedProducts
	wafv2:DescribeManagedProductsByVendor
	wafv2:DescribeManagedRuleGroup
	wafv2:DisassociateWebACL
	wafv2:GenerateMobileSdkReleaseUrl
	wafv2:GetDecryptedAPIKey

Prefixo do serviço	Ações do
	wafv2:GetIPSet
	wafv2:GetLoggingConfiguration
	wafv2:GetManagedRuleSet
	wafv2:GetMobileSdkRelease
	wafv2:GetRateBasedStatementManagedKeys
	wafv2:GetRegexPatternSet
	wafv2:GetRevenueStatistics
	wafv2:GetRevenueStatisticsSummary
	wafv2:GetRevenueStatisticsTimeSeries
	wafv2:GetRuleGroup
	wafv2:GetSampledRequests
	wafv2:GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2:ListAvailableManagedRuleGroupVersions
	wafv2:ListAvailableManagedRuleGroups
	wafv2:ListIPSets
	wafv2:ListLoggingConfigurations
	wafv2:ListManagedRuleSets
	wafv2:ListMobileSdkReleases
	wafv2:ListRegexPatternSets
	wafv2:ListResourcesForWebACL

Prefixo do serviço	Ações do
	wafv2:ListRuleGroups
	wafv2:ListSettlementRecords
	wafv2:ListWebACLs
	wafv2:PutLoggingConfiguration
	wafv2:PutManagedRuleSetVersions
	wafv2:UpdateIPSet
	wafv2:UpdateManagedRuleSetVersionExpiryDate
	wafv2:UpdateRegexPatternSet
	wafv2:UpdateRuleGroup
	wafv2:UpdateWebACL

Prefixo do serviço	Ações do
wellarchitected	wellarchitected:AssociateLenses wellarchitected:AssociateProfiles wellarchitected:CreateLensShare wellarchitected:CreateLensVersion wellarchitected:CreateMilestone wellarchitected:CreateProfile wellarchitected:CreateProfileShare wellarchitected:CreateReviewTemplate wellarchitected:CreateWorkload wellarchitected:CreateWorkloadShare wellarchitected:DeleteLens wellarchitected:DeleteLensShare wellarchitected:DeleteProfile wellarchitected:DeleteProfileShare wellarchitected:DeleteReviewTemplate wellarchitected:DeleteTemplateShare wellarchitected:DeleteWorkload wellarchitected:DeleteWorkloadShare wellarchitected:DisassociateLenses wellarchitected:DisassociateProfiles wellarchitected:ExportLens

Prefixo do serviço	Ações do
	wellarchitected:GetAnswer wellarchitected:GetConsolidatedReport wellarchitected:GetGlobalSettings wellarchitected:GetLens wellarchitected:GetLensReview wellarchitected:GetLensReviewReport wellarchitected:GetLensVersionDifference wellarchitected:GetMilestone wellarchitected:GetProfile wellarchitected:GetProfileTemplate wellarchitected:GetReviewTemplate wellarchitected:GetReviewTemplateAnswer wellarchitected:GetReviewTemplateLensReview wellarchitected:GetWorkload wellarchitected:ImportLens wellarchitected:ListAnswers wellarchitected:ListCheckDetails wellarchitected:ListCheckSummaries wellarchitected:ListLensReviewImprovements wellarchitected:ListLensReviews wellarchitected:ListLensShares

Prefixo do serviço	Ações do
	wellarchitected:ListLenses wellarchitected:ListMilestones wellarchitected:ListNotifications wellarchitected:ListProfileNotifications wellarchitected:ListProfileShares wellarchitected:ListProfiles wellarchitected:ListReviewTemplateAnswers wellarchitected:ListReviewTemplates wellarchitected:ListShareInvitations wellarchitected:ListTemplateShares wellarchitected:ListWorkloadShares wellarchitected:ListWorkloads wellarchitected:UpdateAnswer wellarchitected:UpdateGlobalSettings wellarchitected:UpdateIntegration wellarchitected:UpdateLensReview wellarchitected:UpdateProfile wellarchitected:UpdateReviewTemplate wellarchitected:UpdateReviewTemplateLensReview wellarchitected:UpdateShareInvitation wellarchitected:UpdateWorkload

Prefixo do serviço	Ações do
	wellarchitected:UpdateWorkloadShare wellarchitected:UpgradeLensReview wellarchitected:UpgradeProfileVersion wellarchitected:UpgradeReviewTemplateLensReview

Prefixo do serviço	Ações do
wisdom	wisdom:CreateAssistant
	wisdom:CreateAssistantAssociation
	wisdom:CreateContent
	wisdom:CreateKnowledgeBase
	wisdom:CreateQuickResponse
	wisdom:CreateSession
	wisdom:DeleteAssistant
	wisdom:DeleteAssistantAssociation
	wisdom:DeleteContent
	wisdom:DeleteImportJob
	wisdom:DeleteKnowledgeBase
	wisdom:DeleteQuickResponse
	wisdom:GetAssistant
	wisdom:GetAssistantAssociation
	wisdom:GetContent
	wisdom:GetContentAssociation
	wisdom:GetContentSummary
	wisdom:GetImportJob
	wisdom:GetKnowledgeBase
	wisdom:GetRecommendations
	wisdom:GetSession

Prefixo do serviço	Ações do
	wisdom:ListAssistantAssociations
	wisdom:ListAssistants
	wisdom:ListContentAssociations
	wisdom:ListContents
	wisdom:ListImportJobs
	wisdom:ListKnowledgeBases
	wisdom:ListQuickResponses
	wisdom:NotifyRecommendationsReceived
	wisdom:QueryAssistant
	wisdom:RemoveKnowledgeBaseTemplateUri
	wisdom:SearchContent
	wisdom:SearchQuickResponses
	wisdom:SearchSessions
	wisdom:StartContentUpload
	wisdom:StartImportJob
	wisdom:UpdateContent
	wisdom:UpdateKnowledgeBaseTemplateUri
	wisdom:UpdateQuickResponse
	wisdom:UpdateSession

Prefixo do serviço	Ações do
worklink	worklink:AssociateDomain worklink:AssociateWebsiteAuthorizationProvider worklink:AssociateWebsiteCertificateAuthority worklink:CreateFleet worklink:DeleteFleet worklink:DescribeAuditStreamConfiguration worklink:DescribeCompanyNetworkConfiguration worklink:DescribeDevice worklink:DescribeDevicePolicyConfiguration worklink:DescribeDomain worklink:DescribeFleetMetadata worklink:DescribeIdentityProviderConfiguration worklink:DescribeWebsiteCertificateAuthority worklink:DisassociateDomain worklink:DisassociateWebsiteAuthorizationProvider worklink:DisassociateWebsiteCertificateAuthority worklink:ListDevices worklink:ListDomains worklink:ListFleets worklink:ListWebsiteAuthorizationProviders worklink:ListWebsiteCertificateAuthorities

Prefixo do serviço	Ações do
	worklink:RestoreDomainAccess worklink:RevokeDomainAccess worklink:SignOutUser worklink:UpdateAuditStreamConfiguration worklink:UpdateCompanyNetworkConfiguration worklink:UpdateDevicePolicyConfiguration worklink:UpdateDomainMetadata worklink:UpdateFleetMetadata worklink:UpdateIdentityProviderConfiguration

Prefixo do serviço	Ações do
espaços de trabalho	workspaces:AcceptAccountLinkInvitation
	workspaces:AssociateConnectionAlias
	workspaces:AssociateIpGroups
	workspaces:AssociateWorkspaceApplication
	workspaces:CopyWorkspaceImage
	workspaces:CreateAccountLinkInvitation
	workspaces:CreateConnectClientAddIn
	workspaces:CreateConnectionAlias
	workspaces:CreateIpGroup
	workspaces:CreateStandbyWorkspaces
	workspaces:CreateUpdatedWorkspaceImage
	workspaces:CreateWorkspaceBundle
	workspaces:CreateWorkspaceImage
	workspaces:CreateWorkspaces
	workspaces:CreateWorkspacesPool
	workspaces:DeleteAccountLinkInvitation
	workspaces:DeleteClientBranding
	workspaces:DeleteConnectClientAddIn
	workspaces:DeleteConnectionAlias
	workspaces:DeleteIpGroup
	workspaces:DeleteWorkspaceBundle

Prefixo do serviço	Ações do
	workspaces:DeleteWorkspaceImage
	workspaces:DeployWorkspaceApplications
	workspaces:DeregisterWorkspaceDirectory
	workspaces:DescribeAccount
	workspaces:DescribeAccountModifications
	workspaces:DescribeApplicationAssociations
	workspaces:DescribeApplications
	workspaces:DescribeBundleAssociations
	workspaces:DescribeClientBranding
	workspaces:DescribeClientProperties
	workspaces:DescribeConnectClientAddIns
	workspaces:DescribeConnectionAliasPermissions
	workspaces:DescribeConnectionAliases
	workspaces:DescribeCustomWorkspaceImageImport
	workspaces:DescribeImageAssociations
	workspaces:DescribeIpGroups
	workspaces:DescribeWorkspaceAssociations
	workspaces:DescribeWorkspaceBundles
	workspaces:DescribeWorkspaceDirectories
	workspaces:DescribeWorkspaceImagePermissions
	workspaces:DescribeWorkspaceSnapshots

Prefixo do serviço	Ações do
	workspaces:DescribeWorkspaces
	workspaces:DescribeWorkspacesConnectionStatus
	workspaces:DescribeWorkspacesPoolSessions
	workspaces:DescribeWorkspacesPools
	workspaces:DisassociateConnectionAlias
	workspaces:DisassociateIpGroups
	workspaces:DisassociateWorkspaceApplication
	workspaces:GetAccountLink
	workspaces:ImportClientBranding
	workspaces:ImportWorkspaceImage
	workspaces:ListAccountLinks
	workspaces:ListAvailableManagementCidrRanges
	workspaces:MigrateWorkspace
	workspaces:ModifyAccount
	workspaces:ModifyCertificateBasedAuthProperties
	workspaces:ModifyClientProperties
	workspaces:ModifyEndpointEncryptionMode
	workspaces:ModifySamlProperties
	workspaces:ModifySelfservicePermissions
	workspaces:ModifyStreamingProperties
	workspaces:ModifyWorkspaceAccessProperties

Prefixo do serviço	Ações do
	workspaces:ModifyWorkspaceCreationProperties
	workspaces:ModifyWorkspaceProperties
	workspaces:ModifyWorkspaceState
	workspaces:RebootWorkspaces
	workspaces:RebuildWorkspaces
	workspaces:RegisterWorkspaceDirectory
	workspaces:RejectAccountLinkInvitation
	workspaces:RestoreWorkspace
	workspaces:StartWorkspaces
	workspaces:StartWorkspacesPool
	workspaces:StopWorkspaces
	workspaces:StopWorkspacesPool
	workspaces:TerminateWorkspaces
	workspaces:TerminateWorkspacesPool
	workspaces:TerminateWorkspacesPoolSession
	workspaces:UpdateConnectClientAddIn
	workspaces:UpdateConnectionAliasPermission
	workspaces:UpdateWorkspaceBundle
	workspaces:UpdateWorkspaceImagePermission
	workspaces:UpdateWorkspacesPool

Prefixo do serviço	Ações do
xray	xray:CreateGroup xray:CreateSamplingRule xray:DeleteGroup xray:DeleteResourcePolicy xray:DeleteSamplingRule xray:GetEncryptionConfig xray:GetGroup xray:GetGroups xray:GetInsight xray:GetInsightEvents xray:GetInsightImpactGraph xray:GetInsightSummaries xray:GetSamplingRules xray:ListResourcePolicies xray:PutEncryptionConfig xray:PutResourcePolicy xray:UpdateGroup xray:UpdateSamplingRule

Cotas do IAM Access Analyzer

O IAM Access Analyzer tem as seguintes cotas:

Recurso	Cota padrão	Cota máxima
Máximo de analisadores de nível de conta por tipo de analisador por região da Conta da AWS	1	1
Máximo de analisadores de acesso externo ou não utilizado ao nível organizacional por tipo de analisador por região da Conta da AWS	5	20[1]
Máximo de analisadores de acesso interno ao nível organizacional por organização por região AWS	1	1
O máximo de regras de arquivamento por analisador	100 Cada regra de arquivamento pode ter até 20 valores por critério.	1.000[1]
Número máximo de pré-visualizações de acesso por analisador por hora	1.000	1.000
AWS CloudTrailArquivos de log do processados por gerações de política	100.000	100.000
Gerações simultâneas de política	1	1
Tamanho dos dados de geração de política do AWS CloudTrail	25 GB	25 GB

Recurso	Cota padrão	Cota máxima
Intervalo de tempo de geração de política do AWS CloudTrail	90 dias	90 dias
Gerações de políticas por dia	África (Cidade do Cabo): 5 Ásia-Pacífico (Hong Kong):5 Ásia-Pacífico (Jacarta): 5 Europa (Milão): 5 Oriente Médio (Bahrein): 5 Todas as outras regiões com suporte: 50 **Note** As solicitações de geração de política canceladas se aplicam à cota diária.	África (Cidade do Cabo): 5 Ásia-Pacífico (Hong Kong):5 Ásia-Pacífico (Jacarta): 5 Europa (Milão): 5 Oriente Médio (Bahrein): 5 Todas as outras regiões com suporte: 50

[1]Algumas cotas podem ser configuradas pelo cliente usando o Service Quotas.

Solucionar problemas do IAM

Use as informações contidas aqui para ajudar a diagnosticar e corrigir problemas comuns ao trabalhar com o AWS Identity and Access Management (IAM).

Problemas

- Não consigo fazer login na minha conta da AWS
- Perdi minhas chaves de acesso
- As variáveis da política não estão funcionando
- As alterações que eu faço nem sempre ficam imediatamente visíveis
- Não estou autorizado a executar: iam:DeleteVirtualMFADevice
- Como faço para criar usuários do IAM com segurança?
- Recursos adicionais
- Solucionar problemas de mensagens de erro de acesso negado
- Solucionar problemas de mensagens de erro de acesso recusado com o ID de autorização (versão prévia)
- Solucionar problemas com o usuário-raiz
- Políticas de solução de problemas do IAM
- Solucionar problemas de chaves de acesso e chaves de segurança FIDO
- Solucionar problemas de perfis do IAM
- Solucionar problemas do IAM e Amazon EC2
- Solucionar problemas do IAM e Amazon S3
- Solucionar problemas de federação do SAML com o IAM

Não consigo fazer login na minha conta da AWS

Verifique se você tem as credenciais corretas e se está usando o método correto para fazer login. Para obter mais informações, consulte Solução de problemas de login no Guia do usuário do Início de Sessão da AWS.

Perdi minhas chaves de acesso

As chaves de acesso consistem em duas partes:

- O identificador da chave de acesso. Não se trata de um segredo e pode ser visto no console do IAM sempre que as chaves de acesso forem listadas, como na página de resumo do usuário.
- A chave de acesso secreta da. É fornecida quando você cria inicialmente o par de chaves de acesso. Assim como uma senha, ela não pode ser recuperada posteriormente. Se você perdeu sua chave de acesso secreta, crie um novo par de chaves de acesso. Se você já tiver o número máximo de chaves de acesso, será necessário excluir um par existente antes de criar outro.

Se tiver perdido a chave de acesso secreta, você deverá excluir a chave de acesso e criar uma nova. Para obter mais instruções, consulte Atualizar chaves de acesso.

As variáveis da política não estão funcionando

Se suas variáveis de política não estiverem funcionando, um dos seguintes erros ocorreu:

A data está errada no elemento da política de versão.

Verifique se todas as políticas que incluem variáveis incluem o seguinte número da versão na política: `"Version": "2012-10-17"`. Sem o número da versão correta, as variáveis não são substituídas durante a avaliação. Em vez disso, as variáveis são avaliadas literalmente. Políticas que não incluem variáveis ainda funcionam quando você inclui o número da versão mais recente.

Um elemento de política `Version` é diferente de uma versão de política. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. Uma versão de política é criada quando você modifica uma política gerenciada pelo cliente no IAM. A política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. Para saber mais sobre o elemento de política `Version`, consulte Elementos de política JSON do IAM: Version. Para saber mais sobre as versões de política, consulte the section called “Versionamento de políticas do IAM”.

Os caracteres variáveis estão em maiúsculas e minúsculas erradas.

Verifique se as variáveis de política estão no caso certo. Para obter detalhes, consulte Elementos de política do IAM: variáveis e etiquetas.

As alterações que eu faço nem sempre ficam imediatamente visíveis

Como um serviço que é acessado por meio de computadores em data centers em todo o mundo, o IAM usa um modelo de computação distribuído chamado consistência final. Quaisquer alterações feitas no IAM (ou outros serviços da AWS), incluindo tags de controle de acesso por atributo (ABAC), levam tempo para se tornarem visíveis em todos os endpoints possíveis. Parte do atraso resulta do tempo que leva para enviar dados de servidor para servidor, de zona de replicação para zona de replicação e de região para região. O IAM também usa o armazenamento em cache para melhorar a performance, porém, em alguns casos, isso pode adicionar tempo. A alteração talvez não fique visível enquanto os dados armazenados em cache anteriormente não atingirem o tempo limite.

Você deve projetar seus aplicativos globais levando em conta esses possíveis atrasos. Garanta que eles funcionem conforme o esperado, mesmo quando uma alteração feita em um local não fique imediatamente visível em outro. Essas alterações incluem a criação ou a atualização de usuários, grupos, funções ou políticas. Recomendamos que você não inclua essas alterações do IAM nos caminhos de código críticos e de alta disponibilidade da aplicação. Em vez disso, faça alterações do IAM em uma rotina de inicialização ou de configuração separada que você executa com menos frequência. Além disso, certifique-se de verificar se as alterações foram propagadas antes que os fluxos de trabalho de produção dependam delas.

Para obter mais informações sobre como alguns outros serviços da AWS são afetados por isso, consulte os seguintes recursos:

- Amazon DynamoDB: Consistência de leituras no Guia do desenvolvedor do DynamoDB e Consistência de leituras no Guia do desenvolvedor do Amazon DynamoDB.
- Amazon EC2: Consistência final do EC2 na Referência de API do Amazon EC2.
- Amazon EMR: Ensuring Consistency When Using Amazon S3 and Amazon EMR for ETL Workflows no Blog de big data da AWS
- Amazon Redshift: Gerenciar consistência de dados no Guia do desenvolvedor de banco de dados do Amazon Redshift
- Amazon S3: modelo de consistência de dados do Amazon S3 no Guia do usuário do Amazon Simple Storage Service

Não estou autorizado a executar: iam:DeleteVirtualMFADevice

Você pode receber o seguinte erro ao tentar atribuir ou remover um dispositivo MFA virtual para você ou para outras pessoas:

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to
 perform: iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego
 with an explicit deny
```

Isso pode acontecer se, anteriormente, alguém tiver começado a atribuir um dispositivo com MFA virtual a um usuário no console do IAM e tiver cancelado o processo. Isso cria um dispositivo de MFA virtual para o usuário no IAM, mas nunca o associa a esse usuário. Exclua o dispositivo MFA virtual existente antes de criar um novo com o mesmo nome de dispositivo.

Para corrigir esse problema, um administrador não deve editar permissões de política. Em vez disso, o administrador deve usar a AWS CLI ou a API da AWS para excluir o dispositivo de MFA virtual existente, mas não atribuído.

Para excluir um dispositivo de MFA virtual existente, mas não atribuído

1. Visualize os dispositivos MFA virtuais em sua conta.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API: `ListVirtualMFADevices`
2. Na resposta, localize o ARN do dispositivo de MFA virtual para o usuário que você está tentando corrigir.
3. Exclua o dispositivo de MFA virtual.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - AWS API: `DeleteVirtualMFADevice`

Como faço para criar usuários do IAM com segurança?

Se houver funcionários que precisam de acesso à AWS, você poderá criar usuários do IAM ou usar o IAM Identity Center para autenticação. Se você usa o IAM, a AWS recomenda que você crie um usuário do IAM e informe com segurança as credenciais ao funcionário. Se você não estiver fisicamente localizado ao lado de seu funcionário, use um fluxo de trabalho seguro para comunicar as credenciais aos funcionários.

Use o fluxo de trabalho seguro a seguir para criar um novo usuário no IAM:

1. Crie um novo usuário usando o Console de gerenciamento da AWS. Escolha conceder acesso ao Console de gerenciamento da AWS com uma senha gerada automaticamente. Se necessário, marque a caixa de seleção Users must create a new password at next sign-in (Usuários devem criar uma nova senha no próximo login). Não adicione uma política de permissões ao usuário até que ele tenha alterado sua senha.
2. Depois que o usuário for adicionado, copie o URL de login, o nome de usuário e a senha para o novo usuário. Para visualizar a senha, escolha Show (Mostrar).
3. Envie a senha para seu funcionário usando um método de comunicação segura em sua empresa, como e-mail, chat ou um sistema de emissão de bilhetes. Separadamente, forneça aos usuários o link do console do usuário e o nome de usuário do IAM. Peça que o funcionário confirme se ele pode fazer login com sucesso antes de conceder permissões a ele.
4. Depois que o funcionário confirmar, adicione as permissões que forem necessárias. Como prática recomendada de segurança, adicione uma política que exija que o usuário autentique usando MFA para gerenciar suas credenciais. Para visualizar um exemplo de política, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.

Recursos adicionais

Os seguintes recursos podem ajudar você a solucionar problemas enquanto trabalha com o AWS.

- Guia do usuário do AWS CloudTrail: use o AWS CloudTrail para rastrear um histórico de chamadas de API feitas para a AWS e armazenar essas informações em arquivos de log. Isso ajuda você a determinar quais usuários e contas acessaram recursos na sua conta, quando as chamadas foram feitas, quais ações foram solicitadas, etc. Para obter mais informações, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.
- Centro de Conhecimentos da AWS: encontre perguntas frequentes e links para outros recursos para ajudar na solução de problemas.
- Centro de Suporte da AWS: obtenha suporte técnico.
- Centro de Suporte Premium da AWS: obtenha suporte técnico diferenciado.

Solucionar problemas de mensagens de erro de acesso negado

As informações a seguir podem ajudar você a identificar, diagnosticar e resolver erros de acesso negado com o AWS Identity and Access Management. Erros de acesso negado são exibidos quando a AWS nega explícita ou implicitamente uma solicitação de autorização.

- Uma negação explícita ocorre quando uma política contém uma instrução `Deny` para a ação específica da AWS.
- Uma negação implícita ocorre quando não há nenhuma instrução `Deny` aplicável e também nenhuma instrução `Allow` aplicável. Como uma política do IAM nega uma entidade principal do IAM por padrão, a política deve permitir explicitamente que a entidade principal realize uma ação. Caso contrário, a política nega acesso implicitamente. Para obter mais informações, consulte A diferença entre negações explícitas e implícitas.

Quando você faz uma solicitação a um serviço ou recurso, várias políticas podem ser aplicadas à solicitação. Revise todas as políticas aplicáveis, além da política especificada na mensagem de erro.

- Se várias políticas do mesmo tipo negarem uma solicitação, a mensagem de erro de acesso negado não especificará o número de políticas avaliadas.
- Se vários tipos de políticas negarem uma solicitação de autorização, a AWS só especificará uma dessas políticas na mensagem de erro de acesso negado.

> **Important**
>
> Está com problemas para fazer login na AWS? Certifique-se de estar na página de login da AWS correta para o seu tipo de usuário. Se você for o Usuário raiz da conta da AWS (proprietário da conta), poderá fazer login na AWS usando as credenciais que configurou ao criar a Conta da AWS. Se você for um usuário do IAM, o administrador da sua conta poderá lhe fornecer credenciais de login da AWS. Se você precisar solicitar suporte, não use o link de feedback nesta página. O formulário será recebido pela equipe de documentação da AWS, não do Suporte. Em vez disso, na página Entre em contato conosco, escolha Ainda não consegue fazer login em sua conta da AWS e escolha uma das opções de suporte disponíveis.

Eu recebo a mensagem de “acesso negado” quando faço uma solicitação a um serviço da AWS

- Verifique se a mensagem de erro inclui o tipo e o nome do recurso da Amazon (ARN) da política responsável por negar o acesso. Se for este o caso, verifique se há instruções de recusa para a ação na política especificada. Se o tipo de política for fornecido, mas não houver um ARN de política, concentre-se nas questões de soluções de problemas desse tipo de política: verifique se há instruções de negação para a ação nas políticas do tipo especificado. Se a mensagem de erro não mencionar o tipo de política responsável por negar o acesso, use o restante das diretrizes desta seção para solucionar problemas adicionais.
- Verifique se você tem permissão de política baseada em identidade para chamar a ação e o recurso que solicitou. Se condições forem definidas, você também deverá cumpri-las ao enviar a solicitação. Para obter informações sobre como visualizar ou modificar políticas para um usuário, grupo ou função do IAM, consulte Gerenciar políticas do IAM.
- Se o Console de gerenciamento da AWS retornar uma mensagem informando que você não está autorizado a executar uma ação, entre em contato com o administrador para obter assistência. Seu administrador forneceu a você suas credenciais de login ou link de login.

  O erro de exemplo a seguir ocorre quando o usuário do IAM `mateojackson` tenta usar o console para visualizar detalhes sobre um recurso de *`my-example-widget`* fictício, mas não tem as permissões de `widgets:`*`GetWidget`* fictícias.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  Neste caso, Mateo pede ao administrador para atualizar suas políticas para permitir o acesso ao recurso *`my-example-widget`* usando a ação `widgets:`*`GetWidget`*.
- Você está tentando acessar um serviço que oferece suporte a políticas baseadas em recurso, como o Amazon S3, Amazon SNS ou Amazon SQS? Nesse caso, verifique se a política especifica você como uma entidade principal e lhe concede acesso. Se você fizer uma solicitação para um serviço na sua conta, as políticas baseadas em identidade ou as políticas baseadas em recurso poderão conceder permissão. Se você fizer uma solicitação para um serviço em uma conta diferente, tanto as políticas baseadas em identidade quanto as políticas baseadas em recurso deverão conceder permissão. Para visualizar os serviços que são compatíveis com políticas baseadas em recursos, consulte AWSServiços da que funcionam com o IAM.

- Se sua política incluir uma condição com um par de chave-valor, revise-a com atenção. Os exemplos incluem a chave de condição global `aws:RequestTag/tag-key`, a `kms:EncryptionContext:encryption_context_key` do AWS KMS e a chave de condição `ResourceTag/tag-key` compatível com vários serviços. Certifique-se de que o nome da chave não corresponda a vários resultados. Como os nomes das chaves da condição não diferenciam maiúsculas de minúsculas, uma condição que verifica uma chave chamada `foo` corresponderá a `foo`, `Foo` ou `F00`. Se sua solicitação inclui vários pares de chave-valor com nomes de chaves apenas com a capitalização diferente, o acesso pode ser inesperadamente negado. Para obter mais informações, consulte Elementos de política JSON do IAM: Condition.
- Se você tiver um limite de permissões, verifique se a política usada para o limite de permissões permite sua solicitação. Se suas políticas baseadas em identidade permitirem a solicitação, mas seu limite de permissões não permitir, a solicitação será negada. Um limite de permissões controla o número máximo de permissões que uma entidade principal do IAM (usuário ou perfil) pode ter. As políticas baseadas em recurso não são limitadas pelos limites de permissões. Os limites de permissões não são comuns. Para obter mais informações sobre como a AWS avalia políticas, consulte Lógica da avaliação de política.
- Se você assinar solicitações manualmente (sem usar os AWS SDKs), verifique se você assinou a solicitação corretamente.
- Se você estiver usando uma política de endpoint da Amazon VPC e receber um erro de acesso negado que não esteja registrado em log no AWS CloudTrail, talvez a conta do proprietário do endpoint da VPC seja diferente da conta da chamada ou da conta do perfil de destino.

Eu recebo uma mensagem de "acesso negado" quando faço uma solicitação com credenciais de segurança temporárias

- Primeiro, certifique-se de que não lhe foi negado acesso por um motivo que não esteja relacionado às suas credenciais temporárias. Para obter mais informações, consulte Eu recebo a mensagem de “acesso negado” quando faço uma solicitação a um serviço da AWS.
- Para verificar se o serviço aceita credenciais de segurança temporárias, consulte AWSServiços da que funcionam com o IAM.
- Verifique se suas solicitações estão sendo assinadas corretamente e se a solicitação é bem formada. Para obter mais detalhes, consulte a documentação do toolkit ou Usar credenciais temporárias com recursos da AWS.
- Verifique se suas credenciais de segurança temporárias não expiraram. Para obter mais informações, consulte Credenciais de segurança temporárias no IAM.

- Verifique se o usuário ou a função do IAM tem as permissões corretas. As permissões para credenciais de segurança temporárias são derivadas de um usuário ou uma função do IAM. Como resultado, as permissões são limitadas àquelas que são concedidas à função cujas credenciais temporárias são assumidas. Para obter mais informações sobre como permissões para credenciais de segurança temporárias são determinadas, consulte Permissões de credenciais de segurança temporárias.
- Se você tiver assumido uma função, a sessão da função pode estar limitada por políticas de sessão. Quando solicita credenciais de segurança temporárias de forma programática usando o AWS STS, você pode opcionalmente passar políticas de sessão em linha ou gerenciadas. As políticas de sessão são políticas avançadas que você passa como um parâmetro ao criar uma sessão de credenciais temporárias de forma programática para uma função. É possível passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. É possível usar o parâmetro `PolicyArns` para especificar até 10 políticas de sessão gerenciadas. As permissões da sessão resultante são a interseção das políticas baseadas em identidade da função e das políticas de sessão. Como alternativa, se o administrador ou um programa personalizado fornecer credenciais temporárias a você, ele poderá incluir uma política de sessão para limitar seu acesso.
- Se você for uma entidade principal de usuário federado do AWS STS, a sessão poderá ser limitada pelas políticas de sessão. Você cria uma sessão de usuário federado fazendo login na AWS como um usuário do IAM e, em seguida, solicitando um token de federação. Para obter mais informações, consulte Solicitar credenciais por meio de um intermediador de identidades personalizado. Se você ou seu agente de identidade tiver passado políticas de sessão ao solicitar um token de federação, a sessão será limitada por essas políticas. As permissões da sessão resultantes são a interseção de suas políticas baseadas em identidade do usuário do IAM e as políticas de sessão. Para obter mais informações sobre políticas de sessão, consulte Políticas de sessão.
- Se você estiver acessando um recurso que tenha uma política baseada em recursos usando uma função, verifique se a política concede permissões à função. Por exemplo, a política a seguir permite que `MyRole` da conta `111122223333` acesse `amzn-s3-demo-bucket`.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
```

```
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/*"]
  }]
}
```

Exemplos de mensagens de acesso negado

A maioria das mensagens de erro de acesso negado está no formato `User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. Neste exemplo, o *`user`* é o ARN da entidade principal que não recebe acesso, *`action`* é a ação de serviço que a política nega e *`resource`* é o ARN do recurso em que a política atua. O campo *`context`* fornece contexto adicional sobre o tipo de política que negou o acesso. Em alguns casos, ele também contém o ARN da política que negou o acesso.

Quando uma política nega explicitamente o acesso porque ela contém uma instrução `Deny`, a AWS inclui a frase `with an explicit deny in a` *`type`* `policy` na mensagem de acesso negado. Essa frase também pode especificar o ARN da política, do seguinte modo: `with an explicit deny in a` *`type`* `policy:` *`policy ARN`*.

Quando a política nega acesso implicitamente, a AWS inclui `because no` *`type`* `policy allows the` *`action`* `action` na mensagem de erro de acesso negado.

> **Note**
>
> Alguns serviços da AWS não são compatíveis com esse formato de mensagem de erro de acesso negado. O conteúdo das mensagens de erro de acesso negado pode variar conforme o serviço que está fazendo a solicitação de autorização.

Os exemplos a seguir mostram o formato para diferentes tipos de mensagens de acesso negado.

Acesso negado devido a uma política de controle de serviço: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em suas políticas de controle de serviços (SCPs). No exemplo a seguir, a ação é `codecommit:ListRepositories`.
2. Atualize a SCP adicionando a instrução `Allow`. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS Organizations.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no service control policy allows the codecommit:ListRepositories action
```

Acesso negado devido a uma política de controle de serviço: negação explícita

1. Se um ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação na política de controle de serviços (SCP) especificada. No exemplo abaixo, a ação é `codecommit:ListRepositories`.
2. Se nenhum ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação em suas SCPs.
3. Atualize a SCP removendo a instrução `Deny`. Para obter mais informações, consulte Atualização de uma política de controle de serviços (SCP) no Guia do usuário do AWS Organizations.

Mensagem de erro com um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/service_control_policy/p-
examplepolicyid123
```

Mensagem de erro sem um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
with an explicit deny in a service control policy
```

Acesso negado devido a uma política de controle de recurso: negação explícita

1. Se um ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação na política de controle de recursos (RCP) especificada. No exemplo abaixo, a ação é `secretsmanager:GetSecretValue`.
2. Se nenhum ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação em suas RCPs.
3. Atualize sua RCP removendo a instrução `Deny`. Para obter mais informações, consulte Atualizar uma política de controle de recursos (RCP) no Guia do usuário do AWS Organizations.

Mensagem de erro com um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy:
 arn:aws:organizations::777788889999:policy/o-exampleorgid/resource_control_policy/p-
examplepolicyid456
```

Mensagem de erro sem um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource control policy
```

Acesso negado devido a uma política de endpoint da VPC: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em suas políticas de endpoint da nuvem privada virtual (VPC). No exemplo a seguir, a ação é `codecommit:ListRepositories`.
2. Atualize sua política de endpoint da VPC adicionando a instrução `Allow`. Para obter mais informações, consulte Atualizar uma política de endpoint da VPC no Guia do AWS PrivateLink.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no VPC endpoint policy allows the codecommit:ListRepositories action
```

Acesso negado devido a uma política de endpoint da VPC: negação explícita

1. Verifique se há uma instrução `Deny` explícita para a ação em suas políticas de endpoint da nuvem privada virtual (VPC). No exemplo a seguir, a ação é `codedeploy:ListDeployments`.
2. Atualize sua política de endpoint da VPC removendo a instrução `Deny`. Para obter mais informações, consulte Atualizar uma política de endpoint da VPC no Guia do AWS PrivateLink.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
```

```
with an explicit deny in a VPC endpoint policy
```

Acesso negado devido a um limite de permissões: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em seu limite de permissões. No exemplo a seguir, a ação é `codedeploy:ListDeployments`.
2. Atualize seu limite de permissões adicionando a instrução `Allow` à política do IAM. Para obter mais informações, consulte Limites de permissões para entidades do IAM e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
because no permissions boundary allows the codedeploy:ListDeployments action
```

Acesso negado devido a um limite de permissões: negação explícita

1. Se um ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação no limite de permissões especificado. No exemplo abaixo, a ação é `sagemaker:ListModels`.
2. Se nenhum ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação no limite de permissões anexado à entidade principal.
3. Atualize seu limite de permissões removendo a instrução `Deny` da política do IAM. Para obter mais informações, consulte Limites de permissões para entidades do IAM e Editar políticas do IAM.

Mensagem de erro com um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
with an explicit deny in a permissions boundary: arn:aws:iam::123456789012:policy/
DeveloperPermissionBoundary
```

Mensagem de erro sem um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 sagemaker:ListModels
```

```
with an explicit deny in a permissions boundary
```

Acesso negado devido às políticas de sessão: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em suas políticas de sessão. No exemplo a seguir, a ação é `codecommit:ListRepositories`.
2. Atualize a política de sessão adicionando a instrução `Allow`. Para obter mais informações, consulte Políticas de sessão e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codecommit:ListRepositories
because no session policy allows the codecommit:ListRepositories action
```

Acesso negado devido às políticas de sessão: negação explícita

1. Se um ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação na política de sessão especificada. No exemplo abaixo, a ação é `codedeploy:ListDeployments`.
2. Se nenhum ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação em suas políticas de sessão.
3. Atualize a política de sessão removendo a instrução `Deny`. Para obter mais informações, consulte Políticas de sessão e Editar políticas do IAM.

Mensagem de erro com um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy: arn:aws:iam::123456789012:policy/
DeveloperSessionPolicy
```

Mensagem de erro sem um ARN de política:

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in a session policy
```

Acesso negado devido às políticas baseadas em recursos: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em sua política baseada em recursos. No exemplo a seguir, a ação é `secretsmanager:GetSecretValue`.
2. Atualize a política adicionando a instrução `Allow`. Para obter mais informações, consulte Políticas baseadas em recursos e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
because no resource-based policy allows the secretsmanager:GetSecretValue action
```

Acesso negado devido às políticas baseadas em recursos: negação explícita

1. Verifique se há uma instrução `Deny` explícita para a ação em sua política baseada em recursos. No exemplo a seguir, a ação é `secretsmanager:GetSecretValue`.
2. Atualize a política removendo a instrução `Deny`. Para obter mais informações, consulte Políticas baseadas em recursos e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform:
 secretsmanager:GetSecretValue
on resource: arn:aws:secretsmanager:us-east-1:123456789012:secret:*
with an explicit deny in a resource-based policy
```

Acesso negado devido às políticas de confiança de perfil: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em sua política de confiança de perfil. No exemplo a seguir, a ação é `sts:AssumeRole`.
2. Atualize a política adicionando a instrução `Allow`. Para obter mais informações, consulte Políticas baseadas em recursos e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
because no role trust policy allows the sts:AssumeRole action
```

Acesso negado devido às políticas de confiança de perfil: negação explícita

1. Verifique se há uma instrução `Deny` explícita para a ação em sua política de confiança de perfil. No exemplo a seguir, a ação é `sts:AssumeRole`.
2. Atualize a política removendo a instrução `Deny`. Para obter mais informações, consulte Políticas baseadas em recursos e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:user/John is not authorized to perform: sts:AssumeRole
with an explicit deny in the role trust policy
```

Acesso negado devido a políticas baseadas em identidade: negação implícita

1. Verifique se falta uma instrução `Allow` para a ação em políticas baseadas em identidade anexadas à identidade. No exemplo a seguir, a ação é `codecommit:ListRepositories` anexada ao perfil `HR`.
2. Atualize a política adicionando a instrução `Allow`. Para obter mais informações, consulte Políticas baseadas em identidade e Editar políticas do IAM.

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codecommit:ListRepositories
because no identity-based policy allows the codecommit:ListRepositories action
```

Acesso negado devido às políticas baseadas em identidade: negação explícita

1. Se um ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação na política especificada. No exemplo abaixo, a ação é `codedeploy:ListDeployments`.
2. Se nenhum ARN de política for fornecido na mensagem de erro, verifique se há uma instrução `Deny` para a ação em políticas baseadas em identidade anexadas à identidade.
3. Atualize a política removendo a instrução `Deny`. Para obter mais informações, consulte Políticas baseadas em identidade e Editar políticas do IAM.

Mensagem de erro com um ARN de política:

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
HRAccessPolicy
```

Mensagem de erro sem um ARN de política:

```
User: arn:aws:iam::123456789012:role/HR is not authorized to perform:
 codedeploy:ListDeployments
on resource: arn:aws:codedeploy:us-east-1:123456789012:deploymentgroup:*
with an explicit deny in an identity-based policy
```

Solucionar problemas de mensagens de erro de acesso recusado com o ID de autorização (versão prévia)

> **Important**
>
> O Solucionador de Problemas de Acesso está em versão prévia pública.

Visão geral

Quando a AWS recusa uma solicitação de uma API compatível, a resposta de erro de acesso recusado geralmente inclui um ID de autorização. Esse ID faz referência à avaliação de autorização para essa solicitação recusada. Para recuperar os detalhes da avaliação e identificar o que causou a recusa, faça uma solicitação à API `GetRequestAuthorizationDetails` e forneça o ID de autorização ou abra o link presente na mensagem de erro. Isso permite identificar a causa de uma solicitação recusada em uma única chamada, em vez de resolver as recusas uma por vez.

> **Note**
>
> A AWS não garante um ID de autorização para cada solicitação recusada. Consulte Quando você não receber um ID de autorização neste documento.

O exemplo a seguir mostra uma mensagem de erro de acesso recusado que inclui um ARN de política, um link para o Console de Gerenciamento da AWS e um ID de autorização:

```
User: arn:aws:iam::123456789012:user/janedoe is not authorized to perform:
 iam:CreateInstanceProfile
on resource: arn:aws:iam::123456789012:instance-profile/my-instance-profile
with an explicit deny in an identity-based policy: arn:aws:iam::123456789012:policy/
ExampleDenyPolicy.
Go to https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/
authorization-details/EXAMPLE1a2b3c4d5e6f7g8h9i0j for complete details,
or call the GetRequestAuthorizationDetails API with the following authorization id:
 EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

Conceitos e definições

- Detalhes da autorização. Os detalhes da autorização incluem o contexto da solicitação, as avaliações realizadas e as políticas que foram avaliadas.
- ID de autorização. Um identificador exclusivo incluído em uma resposta de erro de acesso recusado compatível. Ele faz referência à avaliação de autorização para esta solicitação recusada e é o valor que você passa para o link do Console de Gerenciamento da AWS ou para a API `GetRequestAuthorizationDetails`.
- Contexto da solicitação. Os valores das chaves de contexto da condição avaliadas para a solicitação. Isso inclui as chaves de contexto de condição globais e as chaves de contexto de condição específicas do serviço.
- Avaliação. Para cada par de ação/recurso na solicitação, as chaves de contexto de condição aplicadas e o resultado avaliado: `ALLOW`, `EXPLICIT_DENY` (uma política recusou explicitamente a ação) ou `IMPLICIT_DENY` (nenhuma política permitiu a ação) e uma referência a cada política e instrução que correspondia ao par ação/recurso. Cada referência é um ARN de política (para políticas gerenciadas) ou um URI (para políticas em linha e outras políticas não gerenciadas) que é reutilizado em toda a resposta.
- Políticas avaliadas durante a autorização. As políticas da AWS consideradas para a solicitação, incluindo políticas que não correspondem ao par ação/recurso. Cada entrada fornece o tipo de política (por exemplo, baseada em identidade, baseada em recursos, SCP, RCP, limite de permissões ou política de sessão) e o mesmo ARN (para políticas gerenciadas) ou URI usado nas avaliações (para políticas em linha e outras políticas não gerenciadas). Use o ARN ou o URI para combinar uma política com a avaliação que a referenciou. O campo "Anexado a" mostra o ARN principal da entidade principal ao qual a política avaliada está anexada.

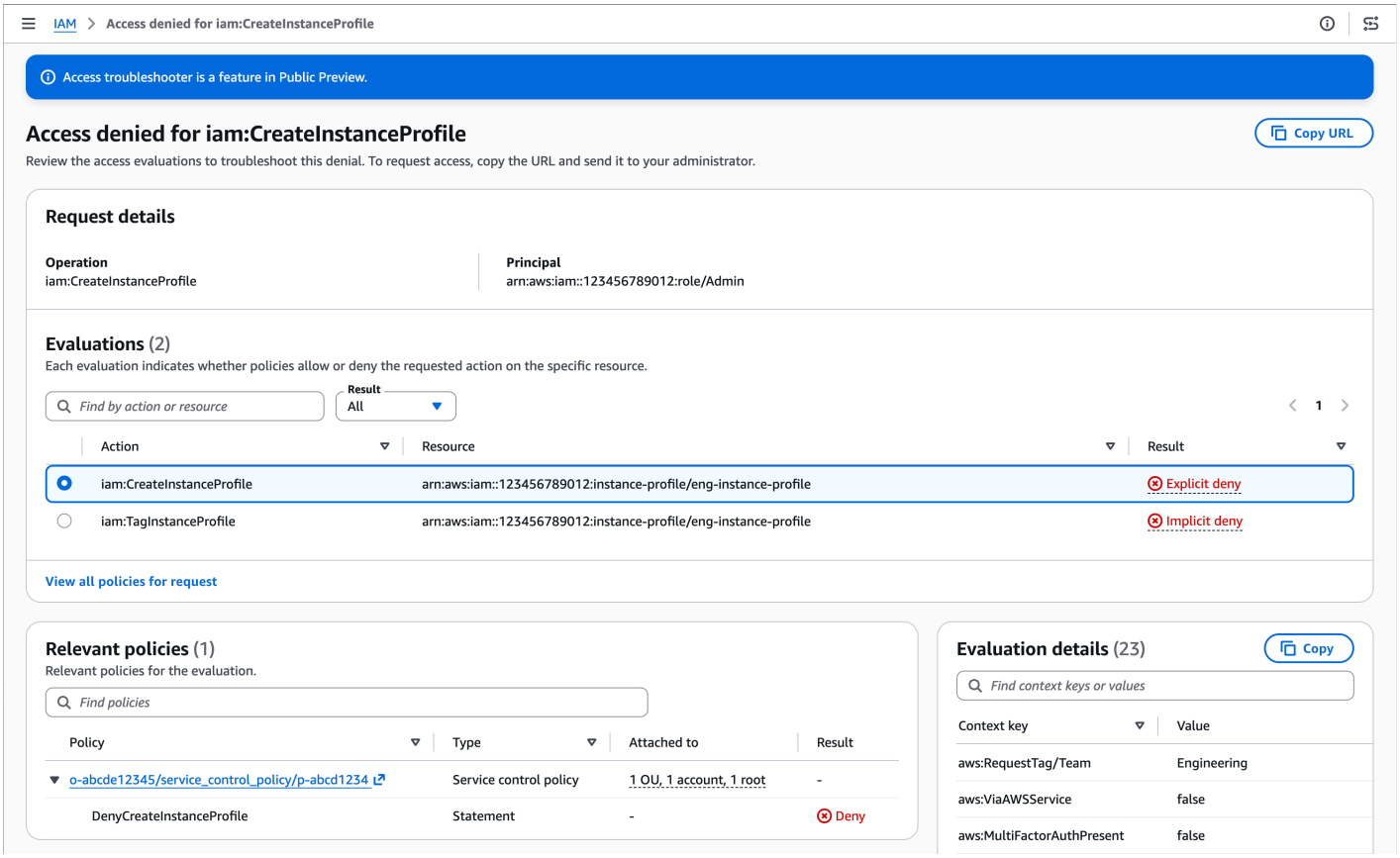
Use esses resultados em conjunto para identificar qual política recusou o acesso e por quê. A avaliação reflete as políticas como estavam no momento da recusa. As políticas podem ter mudado desde então.

A AWS inclui automaticamente um ID de autorização nas respostas de erro de acesso recusado para APIs compatíveis, mas os IDs de autorização não são garantidos. Em alguns casos, a AWS não registra os detalhes da autorização de uma recusa e a resposta de erro não inclui um ID de autorização. Quando os detalhes são registrados, eles podem não estar disponíveis imediatamente, pois a AWS os registra de forma assíncrona. Se uma chamada para `GetRequestAuthorizationDetails` retornar uma resposta não encontrada, aguarde um pouco e tente novamente. A AWS retém os detalhes da autorização por 24 horas após a recusa.

Solucionar problemas usando o Console de Gerenciamento da AWS

Escolha o link na mensagem de erro de acesso recusado para abrir os detalhes da autorização no Console do Solucionador de Problemas de Acesso.

- O que procurar. O Console do Solucionador de Problemas de Acesso mostra a avaliação de cada par de ações/recursos na solicitação, o resultado de cada um (`Allow`, `Explicit deny` ou `Implicit deny`) e as políticas que foram consideradas. Encontre o par que foi recusado e a política que causou a recusa. Consulte a captura de tela a seguir como exemplo:



- Determine se a recusa foi intencional. Revise a avaliação antes de alterar uma política. Uma recusa pode estar correta e, às vezes, a causa é o contexto da solicitação e não a política. Por exemplo, uma solicitação pode não ter uma tag de sessão esperada ou vir de um caminho de rede inesperado.
- Como resolver a recusa. Localize a política correspondente e atualize-a para resolver o problema de acesso. Para uma recusa explícita, remova ou restrinja a instrução `Deny` para a ação. Para uma recusa implícita, adicione uma instrução `Allow` para a ação. Para obter mais informações, consulte Editar políticas do IAM.

Quem pode acessar os detalhes da autorização. Para uma solicitação em uma única conta ou organização, qualquer pessoa que tenha acesso a `iam:GetRequestAuthorizationDetails` pode acessar os detalhes da solicitação. Para uma solicitação que abrange mais de uma organização, um usuário em uma determinada organização pode visualizar somente os detalhes que dizem respeito à sua própria organização.

Solução de problemas usando a API ou a AWS CLI da AWS

Para solucionar uma recusa usando a API ou a AWS CLI da AWS:

1. Localize o ID de autorização na resposta de erro de acesso recusado.
2. Chame `GetRequestAuthorizationDetails` com o ID de autorização, na mesma região Região da AWS em que a recusa ocorreu. O exemplo a seguir mostra o comando da AWS CLI:

```
aws iam-toolbox get-request-authorization-details \
    --region us-east-1 \
    --authorization-id EXAMPLE1a2b3c4d5e6f7g8h9i0j
```

3. Analise os resultados da avaliação para identificar quais políticas recusaram o acesso e o motivo da recusa.
4. Atualize as políticas identificadas para resolver o problema de acesso. Para obter mais informações, consulte Editar políticas do IAM.

Quando você não receber um ID de autorização

A AWS não retorna um ID de autorização nos seguintes casos:

- A autorização é muito grande para ser registrada. Quando uma solicitação envolve um grande número de políticas, a AWS pode decidir não registrar a solicitação.
- O serviço ainda não é compatível. Consulte Serviços com suporte.
- A API não é compatível. Algumas APIs em um serviço compatível não retornam um ID de autorização.
- O tipo de solicitação não é compatível. Isto inclui solicitações feitas por um serviço em seu nome e solicitações anônimas.

Se você não receber um ID de autorização, consulte Solucionar problemas de mensagens de erro de acesso negado para outras formas de diagnosticar a recusa.

Considerações adicionais

Permissões obrigatórias. Para chamar `GetRequestAuthorizationDetails`, você deve ter permissões para a ação `iam:GetRequestAuthorizationDetails`.

API regional. `GetRequestAuthorizationDetails` é uma API regional. Chame-a na mesma região Região da AWS em que o erro de acesso recusado e o ID de autorização foram gerados.

Solicitações entre organizações. Quando uma solicitação recusada envolve mais de uma organização, `GetRequestAuthorizationDetails` retorna somente os detalhes que pertencem à sua própria organização. Isso se aplica em ambas as direções: se você estiver na organização do chamador recusado, verá somente os detalhes relacionados a essa organização. Se você estiver na organização do recurso recusado, verá somente os detalhes que dizem respeito a essa organização. Detalhes pertencentes a outra organização, como as políticas ou os valores de chave de contexto desta organização, não são retornados.

Se você receber um erro de acesso recusado de uma API, serviço ou região que ainda não inclui um ID de autorização, consulte Solucionar problemas de mensagens de erro de acesso negado.

Serviços com suporte

O suporte para `GetRequestAuthorizationDetails` está sendo implementado gradualmente em APIs da AWS e na região Região da AWS. O(s) seguinte(s) serviço(s) oferece(m) suporte aos `GetRequestAuthorizationDetails`:

- A maioria das APIs no IAM

Solucionar problemas com o usuário-raiz

Use as informações aqui contidas para obter ajuda para solucionar problemas relacionados ao usuário-raiz de uma Conta da AWS.

> **Note**
>
> Contas da AWS gerenciadas por meio do AWS Organizations podem ter acesso raiz centralizado habilitado para contas-membro. Essas contas-membro não têm credenciais de usuário-raiz, não podem fazer login como usuário-raiz e são impedidas de recuperar a senha do usuário-raiz. Entre em contato com seu administrador se precisar realizar uma tarefa que exija credenciais de usuário-raiz.

Não consigo realizar as tarefas que espero poder realizar quando estou conectado como usuário-raiz da conta

A conta pode ser membro de uma organização no AWS Organizations. O administrador da sua organização pode ter uma política de controle de serviços (SCP) para limitar as permissões da

sua conta. As SCPs afetam todos os usuários, incluindo o usuário-raiz. Para saber mais, consulte Políticas de controle de serviço no Guia do usuário do AWS Organizations.

Esqueci a senha de usuário-raiz da minha Conta da AWS

Se você é um usuário-raiz e perdeu ou esqueceu a senha da sua Conta da AWS, é possível redefinir essa senha. É necessário saber o endereço de e-mail usado para criar a Conta da AWS e ter acesso à conta de email. Para obter mais informações, consulte Redefinir uma senha de usuário-raiz perdida ou esquecida.

Não tenho acesso ao e-mail da minha Conta da AWS

Ao criar uma Conta da AWS, você fornece um endereço de e-mail e uma senha. Essas são as credenciais para o Usuário raiz da conta da AWS. Se você não tiver certeza do endereço de e-mail associado à Conta da AWS, pesquise por mensagens enviadas por `@signin.aws` ou `@verify.signin.aws` a qualquer endereço de e-mail da sua organização que possa ter sido usado para abrir a Conta da AWS.

Se você souber o endereço de e-mail, mas não tiver mais acesso a ele, tente recuperar o acesso. Use uma das seguintes opções para recuperar o acesso ao seu e-mail:

- Se você for o proprietário do domínio do endereço de e-mail, poderá restaurar um endereço de e-mail excluído. Como alternativa, você pode configurar um catch-all para sua conta de e-mail. Um catch-all coleta todas as mensagens enviadas para endereços de e-mail que não existem mais no servidor de e-mail e as redireciona para outro endereço de e-mail.
- Se o endereço de e-mail da conta é parte do seu sistema de e-mail corporativo, recomendamos que você entre em contato com os administradores do sistema de TI. Eles podem ajudar você a obter acesso novamente ao e-mail.

Se você ainda não conseguir fazer login na sua Conta da AWS, encontre opções alternativas de suporte em Entre em contato conosco.

Políticas de solução de problemas do IAM

Uma política é uma entidade da AWS que, quando associada a uma identidade ou recurso, define suas permissões. A AWS avalia essas políticas quando uma principal, como um usuário, faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada.

As políticas são armazenadas na AWS como documentos JSON anexados a entidades principais como políticas baseadas em identidade ou a recursos como políticas baseadas em recursos. As políticas baseadas em identidade incluem as políticas gerenciadas pela AWS, as políticas gerenciadas pelo cliente e as políticas em linha.

Você pode criar e editar políticas gerenciadas pelo cliente no Console de gerenciamento da AWS usando ambas as opções de editor Visual e JSON. Quando você visualiza uma política no Console de gerenciamento da AWS, pode ver um resumo das permissões concedidas por essa política. Você pode usar o editor visual e os resumos das políticas para ajudar a diagnosticar e corrigir os erros comuns encontrados no gerenciamento das políticas do IAM.

Lembre-se de que todas as políticas do IAM são armazenadas usando uma sintaxe que começa com as regras da Notação de objetos JavaScript (JSON). Não é preciso entender esta sintaxe para criar ou gerenciar políticas. É possível criar e editar uma política usando o editor visual no Console de gerenciamento da AWS. Para saber mais sobre a sintaxe JSON nas políticas do IAM, consulte Gramática da linguagem das políticas de JSON do IAM.

Solução de problemas de tópicos das políticas do IAM

- Solução de problemas usando o editor visual
  - Reestruturação da política
  - Escolher um ARN de recurso no editor visual
  - Negar permissões no editor visual
  - Especificar vários serviços no editor visual
  - Reduzir o tamanho da política no editor visual
  - Corrigir serviços, ações ou tipos de recursos desconhecidos no editor visual
- Solucionar problemas com resumos de políticas
  - Resumo de política ausente
  - O resumo de política inclui serviços, ações ou tipos de recursos desconhecidos
  - O serviço não oferece suporte a resumos de política do IAM
  - Minha política não concede as permissões esperadas
- Solução de problemas de gerenciamento de políticas
  - Anexar ou desanexar uma política em uma conta do IAM
  - Alterar políticas de suas identidades do IAM com base em sua atividade
- Solução de problemas de documentos de políticas JSON

- Validar suas políticas
- Não tenho permissões para validação de política no editor JSON
- Mais de um objeto de política JSON
- Mais de um elemento de instrução JSON
- Mais de um efeito, ação ou elemento de recurso em um elemento de instrução JSON
- Elemento de versão JSON ausente

Solução de problemas usando o editor visual

Ao criar ou editar uma política gerenciada pelo cliente, você pode usar as informações no editor Visual para ajudá-lo a solucionar erros na política. Para visualizar um exemplo de como usar o editor visual para criar uma política, consulte the section called “Controle de acesso a identidades”.

Reestruturação da política

Quando você cria uma política, a AWS valida, processa e transforma a política antes de armazená-la. Quando a política é recuperada, a AWS a transforma de volta para um formato legível por humanos sem alterar as permissões. Isso pode resultar em diferenças no que diz respeito ao que você vê no editor visual de política ou na guia JSON.

- Blocos de permissões do editor visual podem ser adicionados, removidos ou reorganizados, e o conteúdo dentro de um bloco pode ser otimizado.
- Na guia JSON, espaços em branco insignificantes podem ser removidos e os elementos dentro dos mapas JSON podem ser reordenados. Além disso, os IDs da Conta da AWS nos elementos de entidade principal podem ser substituídos pelo nome do recurso da Amazon (ARN) do Usuário raiz da conta da AWS.

Devido a essas possíveis alterações, você não deve comparar documentos de políticas JSON como strings.

Ao criar uma política gerenciada pelo cliente no Console de gerenciamento da AWS, você pode optar por trabalhar exclusivamente no editor JSON. Se você nunca alterar a política no editor Visual e escolher Avançar no editor JSON, a probabilidade de que a política seja reestruturada é menor. Ao usar o editor Visual, o IAM pode reestruturar a política para otimizar a aparência. Essa reestruturação existe somente na sua sessão de edição e não é salva automaticamente.

Se a sua política for reestruturada em uma sessão de edição, o IAM determinará se a reestruturação deve ser salva com base nas seguintes situações:

Usar essa opção de editor	Se você editar a política	E depois escolher Avançar nesta guia	Quando você escolher Salvar alterações
Visual	Editada	Visual	A política será reestruturada
Visual	Editada	JSON	A política será reestruturada
Visual	Não editada	Visual	A política será reestruturada
JSON	Editada	Visual	A política será reestruturada
JSON	Editada	JSON	A estrutura da política não será alterada
JSON	Não editada	JSON	A estrutura da política não será alterada

O IAM pode reestruturar políticas complexas ou políticas que têm blocos de permissões ou instruções que permitam vários serviços, tipos de recursos ou chaves de condição.

Escolher um ARN de recurso no editor visual

Ao criar ou editar uma política usando o editor visual, você deve primeiro escolher um serviço e, em seguida, selecionar as ações do serviço. Se o serviço e as ações que você selecionou oferecerem suporte à escolha de recursos específicos, o editor visual apresentará uma lista dos tipos de recurso compatíveis. Em seguida, você poderá escolher Adicionar ARN para fornecer detalhes sobre seu recurso. Você pode escolher as opções a seguir para adicionar um ARN a um tipo de recurso.

- Uso do builder de ARN: dependendo do tipo de recurso, você pode ver campos diferentes para construir seu ARN. Você também pode escolher Qualquer para fornecer permissão para qualquer valor na configuração especificada. Por exemplo, se você tiver selecionado o grupo do nível de

acesso Read (Leitura) do Amazon EC2, as ações da política oferecerão suporte ao tipo de recurso `instance`. Forneça os valores Região, Conta e InstanceId para o recurso. A política concede permissões para qualquer instância na sua conta se você fornecer o ID da conta e escolher Qualquer para a região e ID da instância.

- Digitar ou colar o ARN: você pode especificar recursos pelo nome do recurso da Amazon (ARN). Você pode incluir um caractere curinga (*****) em qualquer campo do ARN entre cada par de dois-pontos. Para obter mais informações, consulte Elementos de política JSON do IAM: Resource.

Negar permissões no editor visual

Por padrão, uma política criada usando o editor visual permite as ações que você escolhe. Para negar as ações escolhidas, selecione Alternar para negar permissões. Como as solicitações são negadas por padrão, recomendamos que você conceda permissões somente às ações e aos recursos dos quais um usuário precisa. Você só precisa criar uma instrução de negação se quiser substituir uma permissão concedida por outra instrução ou política. Recomendamos que você limite ao mínimo o número de permissões de negação, pois elas podem aumentar a dificuldade de solucionar problemas nas permissões. Para obter mais informações sobre como o IAM avalia a lógica de políticas, consulte Lógica da avaliação de política.

> **Note**
>
> Por padrão, somente o Usuário raiz da conta da AWS tem acesso a todos os recursos na conta. Portanto, se você não estiver conectado como usuário raiz, você deverá ter as permissões concedidas por uma política.

Especificar vários serviços no editor visual

Quando você usa o editor visual para criar uma política, só pode selecionar um serviço por vez. Esta é uma prática recomendada, pois o editor visual permite que você escolha as ações para esse serviço específico. Em seguida, você escolhe os recursos compatíveis com esse serviço e as ações selecionadas. Isso facilita a criação e a solução de problemas na sua política.

Você também pode usar um caractere curinga (*) para especificar manualmente vários serviços. Por exemplo, digite **`Code*`** para fornecer permissões para todos os serviços que comecem com `Code`, como `CodeBuild` e `CodeCommit`. No entanto, você deve, em seguida, inserir as ações e os ARNs dos recursos para concluir sua política. Além disso, quando você salvar a política, ela poderá ser reestruturada a fim de incluir cada serviço em um bloco de permissões separado.

Ou então, para usar a sintaxe JSON (como caracteres curinga) para serviços, crie, edite e salve a política usando a opção de editor JSON.

Reduzir o tamanho da política no editor visual

Quando você usa o editor visual para criar uma política, o IAM cria um documento JSON para armazenar sua política. Você pode visualizar este documento mudando para a opção de editor JSON. Se este documento JSON exceder o limite de tamanho de uma política, o editor visual exibirá uma mensagem de erro. Você não poderá revisar e salvar a política. Para visualizar a limitação do IAM quanto ao tamanho de uma política gerenciada, consulte Limites de caracteres do IAM e do STS.

Para reduzir o tamanho de uma política no editor visual, edite a política ou mova blocos de permissões para outra política. A mensagem de erro inclui o número de caracteres que o documento da política contém. Você pode usar essas informações para ajudar a reduzir o tamanho da política.

Corrigir serviços, ações ou tipos de recursos desconhecidos no editor visual

Você pode ver um aviso no editor visual de que sua política inclui um serviço, ação ou tipo de recurso não reconhecido.

> **Note**
>
> O IAM revisa nomes de serviço, ações e tipos de recurso para serviços que oferecem suporte a resumos de políticas. Contudo, seu resumo de política pode incluir um valor de recurso ou uma condição que não existe. Sempre teste as políticas com o simulador de políticas.

Se a sua política inclui serviços, ações ou tipos de recurso desconhecidos, ocorreu um dos seguintes erros:

- Serviço de pré-visualização: serviços que estão em pré-visualização não oferecem suporte ao editor visual. Se estiver participando da pré-visualização, você deverá inserir manualmente as ações e os ARNs dos recursos para concluir sua política. Você pode ignorar qualquer aviso e continuar. Ou então, você pode escolher a opção de editor JSON e digitar ou colar um documento de política JSON.
- Serviço personalizado: serviços personalizados não oferecem suporte ao editor visual. Se estiver usando um serviço personalizado, você deverá inserir manualmente as ações e os ARNs dos

recursos para concluir sua política. Você pode ignorar qualquer aviso e continuar. Ou então, você pode escolher a opção de editor JSON e digitar ou colar um documento de política JSON.

- O serviço não é compatível com o editor visual: se sua política incluir um serviço em disponibilidade geral (GA) que não seja compatível com o editor visual, você deverá inserir manualmente as ações e os ARNs dos recursos para concluir sua política. Você pode ignorar qualquer aviso e continuar. Ou então, você pode escolher a opção de editor JSON e digitar ou colar um documento de política JSON.

  Os serviços disponíveis são serviços que são lançados publicamente e não são serviços personalizados ou de visualização. Se o serviço desconhecido for um serviço disponível para o público geral e seu nome estiver digitado corretamente, isso significa que o serviço não oferece suporte ao editor visual. Para saber como solicitar suporte a um resumo de políticas ou ao editor visual para um serviço disponível para o público, consulte O serviço não oferece suporte a resumos de política do IAM.
- A ação não é compatível com o editor visual: se sua política incluir um serviço compatível com uma ação incompatível, você deverá inserir manualmente as ações e os ARNs dos recursos para concluir sua política. Você pode ignorar qualquer aviso e continuar. Ou então, você pode escolher a opção de editor JSON e digitar ou colar um documento de política JSON.

  Se a política incluir um serviço com suporte com uma ação sem suporte, isso significa que o serviço não oferece suporte completo ao editor visual. Para saber como solicitar suporte a um resumo de políticas ou ao editor visual para um serviço disponível para o público, consulte O serviço não oferece suporte a resumos de política do IAM.
- O tipo de recurso não oferece suporte ao editor visual: se a política incluir uma ação com suporte com um tipo de recurso sem suporte, você poderá ignorar o aviso e prosseguir. No entanto, como o IAM não pode confirmar se você incluiu recursos para todas as ações selecionadas, você poderá receber avisos adicionais.
- Erro de digitação: quando você insere manualmente um serviço, uma ação ou um recurso no editor visual, é possível criar uma política que inclua um erro de digitação. Recomendamos usar o editor visual selecionando os serviços e ações da lista. Em seguida, preencha a seção de recursos de acordo com os prompts. Se um serviço não for totalmente compatível com o editor visual, talvez você tenha de inserir partes da política manualmente.

  Se você tiver certeza de que sua política não contém nenhum dos erros acima, ela poderá incluir um erro de digitação. Verifique se há os seguintes problemas:

- Nomes de serviço, ação e tipo de recurso com erros de digitação, como `s2` em vez de `s3` ou `ListMyBuckets` em vez de `ListAllMyBuckets`
- Texto desnecessário em ARNs, como `arn:aws:s3: : :*`
- Dois pontos ausentes em ações, como `iam.CreateUser`

Você pode avaliar uma política que pode incluir erros de digitação selecionando Avançar para revisar o resumo da política. Em seguida, confirme se a política fornece as permissões que você espera.

Solucionar problemas com resumos de políticas

Você pode diagnosticar e resolver problemas relacionados aos resumos de políticas.

Resumo de política ausente

O console do IAM inclui tabelas do resumo de políticas que descrevem o nível de acesso, os recursos e as condições permitidas ou negadas para cada serviço em uma política. As políticas são resumidas em três tabelas: o resumo de políticas, o resumo de serviços e o resumo de ações.

A tabela de resumo da política inclui uma lista de serviços e resumos das permissões que são definidas pela política escolhida. O resumo da política está disponível na página Detalhes da política para qualquer política anexada ou na página Políticas para políticas gerenciadas. Se a AWS não puder renderizar um resumo para uma política, o console exibirá o documento da política JSON e o seguinte erro:

Não é possível gerar um resumo para esta política. Você ainda pode visualizar ou editar o documento da política JSON.

Se a política não incluir um resumo, terá ocorrido um dos seguintes erros:

- Elemento de política sem suporte: o IAM não oferece suporte à geração de resumos de políticas para políticas que incluem um dos seguintes elementos de política:
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Nenhuma permissão de política: se uma política não fornecer permissões efetivas, o resumo da política não poderá ser gerado. Por exemplo, se uma política inclui uma única instrução

com o elemento `"NotAction": "*"`, ela concede acesso a todas as ações, exceto "todas as ações" (*). Isso significa que ela `Deny` ou `Allow` acesso a nada.

> **Note**
>
> Tenha cuidado ao usar esses elementos de políticas, como `NotPrincipal`, `NotAction` e `NotResource`. Para obter mais informações sobre o uso de elementos de política, consulte Referência de elemento de política JSON do IAM.

Se você fornecer serviços e recursos incompatíveis, poderá criar uma política que não concede permissões eficazes. Isso pode ocorrer quando você especifica ações em um serviço e recursos de outro serviço. Nesse caso, o resumo de políticas aparecerá. A única indicação de que há um problema é que a coluna de recursos no resumo pode incluir um recurso de um serviço diferente. Se essa coluna incluir um recurso incompatível, será necessário analisar sua política para busca de erros. Teste suas políticas com o simulador de políticas para melhor entendê-las.

O resumo de política inclui serviços, ações ou tipos de recursos desconhecidos

No console do IAM, se um resumo de política incluir um símbolo de aviso
( ),
a política poderá incluir um serviço, ação ou tipo de recurso desconhecido. Para saber mais sobre os avisos de um resumo de políticas, consulte Resumo da política (lista de serviços).

> **Note**
>
> O IAM revisa nomes de serviço, ações e tipos de recurso para serviços que oferecem suporte a resumos de políticas. Contudo, seu resumo de política pode incluir um valor de recurso ou uma condição que não existe. Sempre teste as políticas com o simulador de políticas.

Se a sua política inclui serviços, ações ou tipos de recurso desconhecidos, ocorreu um dos seguintes erros:

- Serviço de pré-visualização: serviços que estão em pré-visualização não oferecem suporte a resumos de política.

- Serviço personalizado: serviços personalizados não oferecem suporte a resumos de política.
- O serviço não oferece suporte a resumos: se a política incluir um serviço disponível para o público (GA) que não ofereça suporte a resumos de política, o serviço é incluído na seção Unrecognized services (Serviços desconhecidos) da tabela do resumo da política. Os serviços disponíveis são serviços que são lançados publicamente e não são serviços personalizados ou de visualização. Se um serviço desconhecido estiver disponível para o público e o nome estiver escrito corretamente, o serviço não oferecerá suporte a resumos de política do IAM. Para saber como solicitar suporte para um resumo de políticas para um serviço disponível, consulte O serviço não oferece suporte a resumos de política do IAM.
- A ação não oferece suporte a resumos: se a política incluir um serviço compatível com uma ação incompatível, a ação será incluída na seção Unrecognized actions (Ações desconhecidas) da tabela do resumo do serviço. Para saber mais sobre os avisos de um resumo de serviços, consulte Resumo do serviço (lista de ações).
- O tipo de recurso não oferece suporte a resumos: se a política incluir uma ação compatível com um tipo de recurso incompatível, o recurso será incluído na seção Unrecognized resource types (Tipos de recurso desconhecidos) da tabela do resumo do serviço. Para saber mais sobre os avisos de um resumo de serviços, consulte Resumo do serviço (lista de ações).
- Erro de digitação: a AWS verifica se o JSON está sintaticamente correto e se a política não inclui erros de digitação ou outros erros como parte da validação de política.

> **Note**
>
> Como prática recomendada, recomendamos que você use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais. Recomendamos que você abra as políticas existentes e analise e resolva todas as recomendações de validação de política.

O serviço não oferece suporte a resumos de política do IAM

É possível que resumos de políticas do IAM ou o editor visual não sejam compatíveis com uma ação ou um serviço em disponibilidade geral (GA). Os serviços disponíveis são serviços lançados publicamente e não serviços personalizados ou de visualização. Se o serviço desconhecido for um serviço disponível para o público e seu nome estiver digitado corretamente, então o serviço não oferece suporte a esses recursos. Se a política incluir um serviço compatível com uma ação incompatível, o serviço não oferecerá suporte completo a resumos de política do IAM.

Para solicitar que um serviço adicione suporte ao resumo de política do IAM ou ao editor visual

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Localize a política que inclui o serviço sem suporte:
   - Se a política for uma política gerenciada, escolha Políticas no painel de navegação. Na lista de políticas, escolha o nome da política que deseja visualizar.
   - Se a política for uma política em linha anexada ao usuário, escolha Usuários no painel de navegação. Na lista de usuários, escolha o nome do usuário cuja política deseja visualizar. Na tabela de políticas do usuário, expanda o cabeçalho do resumo de políticas que deseja visualizar.
3. No lado esquerdo do rodapé do Console de gerenciamento da AWS, escolha Comentários. Na caixa Feedback para o IAM, digite **`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**. Se quiser que mais de um serviço ofereça suporte aos resumos, digite **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

Para solicitar que um serviço adicione suporte ao resumo de política do IAM para uma ação ausente

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Localize a política que inclui o serviço sem suporte:
   - Se a política for uma política gerenciada, escolha Políticas no painel de navegação. Na lista de políticas, escolha o nome da política que deseja visualizar.
   - Se a política for uma política em linha anexada ao usuário, escolha Usuários no painel de navegação. Na lista de usuários, escolha o nome do usuário cuja política deseja visualizar. Na tabela de políticas do usuário, escolha o nome da política que deseja visualizar para expandir o resumo de políticas.
3. No resumo de políticas, escolha o nome do serviço que inclua uma ação sem suporte.
4. No lado esquerdo do rodapé do Console de gerenciamento da AWS, escolha Comentários. Na caixa Feedback para o IAM, digite **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**. Se quiser relatar mais de uma ação sem suporte, digite **`I request that the`**

<ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions.

Para solicitar que outro serviço inclua ações ausentes, repita as três últimas etapas.

Minha política não concede as permissões esperadas

Para atribuir permissões a um usuário, grupo, função ou recurso, você cria uma política, que é um documento que define as permissões. Um documento de política inclui os seguintes elementos:

- Effect (Efeito): se a política permite ou nega acesso
- Action (Ação): a lista de ações permitidas ou negadas pela política
- Resource (Recurso): a lista de recursos nos quais as ações podem ocorrer
- Condition (Condição) (opcional): as circunstâncias sob as quais a política concede a permissão

Para saber mais sobre este e outros elementos de política, consulte Referência de elemento de política JSON do IAM.

Para conceder acesso, sua política deve definir uma ação com um recurso compatível. Se a política também inclui uma condição, essa condição deve incluir uma chave de condição global ou deve se aplicar à ação. Para saber quais recursos são compatíveis com uma ação, consulte a documentação da AWS para o seu serviço. Para saber quais condições são compatíveis com uma ação, consulte Ações, recursos e chaves de condição de serviços da AWS.

Verifique se sua política define uma ação, recurso ou condição que não concede permissões. Veja o resumo da política de sua política usando o console do IAM em https://console.aws.amazon.com/iam/. Você pode usar resumos de políticas para identificar e corrigir problemas em sua política.

Há vários motivos pelos quais um elemento pode não conceder permissões, apesar de ser definido na política do IAM:

- Uma ação é definida sem um recurso aplicável
- Um recurso é definido sem uma ação aplicável
- Uma condição é definida sem uma ação aplicável

Para ver exemplos de resumos de política que incluem avisos, consulte the section called “Resumo da política (lista de serviços)”.

Uma ação é definida sem um recurso aplicável

A política abaixo define todas as ações `ec2:Describe*` e define um recurso específico. Nenhuma das ações `ec2:Describe` é concedida porque nenhuma dessas ações é compatível com as permissões no nível de serviço. Permissões no nível de recurso significam que a ação é compatível com recursos que usam ARNs no elemento `Resource` da política. Se uma ação não é compatível com as permissões no nível de recurso, a instrução na política deve usar um curinga (*) no elemento `Resource`. Para saber quais serviços da são compatíveis com permissões em nível do recurso, consulte AWSServiços da que funcionam com o IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "arn:aws:ec2:us-east-2:111122223333:instance/*"
        }
    ]
}
```

Esta política não fornece permissões, e o resumo da política inclui o seguinte erro:

`This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Para corrigir essa política, você deve usar * no elemento `Resource`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
```

```
}
```

Um recurso é definido sem uma ação aplicável

A política abaixo define um recurso de bucket do Amazon S3, mas não inclui uma ação do S3 que pode ser executada nesse recurso. Esta política também concede acesso completo a todas as ações do Amazon CloudFront.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
        }
    ]
}
```

Esta política fornece permissões para todas as ações do CloudFront. Mas como a política define o recurso do S3 `amzn-s3-demo-bucket` sem definir ações do S3, o resumo da política inclui o seguinte aviso:

`This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Para corrigir essa política para fornecer as permissões de bucket do S3, você deve definir as ações do S3 que podem ser executadas em um recurso de bucket.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudfront:*",
                "s3:CreateBucket",
                "s3:ListBucket*",
                "s3:PutBucket*",
                "s3:GetBucket*"
            ],
            "Resource": [
                "arn:aws:cloudfront:*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

Como alternativa, para corrigir essa política para fornecer apenas as permissões do CloudFront, remova o recurso do S3.

Uma condição é definida sem uma ação aplicável

A política abaixo define duas ações do Amazon S3 para todos os recursos do S3, se o prefixo do S3 for igual a `custom` e o ID da versão for igual a `1234`. No entanto, a chave de condição `s3:VersionId` é usada para a marcação da versão do objeto e não é suportada pelas ações do bucket definidas. Para saber quais condições são compatíveis com uma ação, consulte Actions, Resources, and Condition Keys for AWS Services e escolha o serviço para visualizar a documentação do serviço sobre chaves de condição.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
```

```
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Esta política fornece permissões para a ação `s3:ListBucketVersions` e a ação `s3:ListBucket` se o nome do bucket inclui o prefixo `custom`. Mas, como a condição `s3:VersionId` não é compatível com nenhuma das ações definidas, o resumo da política inclui o seguinte erro:

```
This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.
```

Para corrigir essa política para usar a identificação de versão do objeto do S3, você deve definir uma ação do S3 compatível com a chave de condição `s3:VersionId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Esta política fornece permissões para cada ação e condição na política. No entanto, a política ainda não fornece permissões, pois não há casos em que uma única ação corresponde a ambas as condições. Em vez disso, você deve criar duas instruções separadas e cada uma deve incluir apenas as ações com as condições às quais se aplicam.

Para corrigir essa política, crie duas instruções. A primeira declaração inclui as ações que suportam a condição `s3:prefix` e a segunda instrução inclui as ações que suportam a condição `s3:VersionId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Solução de problemas de gerenciamento de políticas

Você pode diagnosticar e resolver problemas relacionados ao gerenciamento de políticas.

Anexar ou desanexar uma política em uma conta do IAM

Algumas políticas gerenciadas da AWS estão vinculadas a um serviço. Essas políticas são usadas apenas com uma função vinculada ao serviço desse serviço. No console do IAM, quando você visualiza a página Detalhes da política, a página inclui um banner para indicar que a política está vinculada a um serviço. Não é possível anexar essa política a um usuário, grupo ou função no IAM. Quando você cria uma função vinculada ao serviço para o serviço, essa política é automaticamente anexada a sua nova função. Como a política é necessária, não é possível desanexar a política da função vinculada ao serviço.

Alterar políticas de suas identidades do IAM com base em sua atividade

Você pode atualizar as políticas para suas identidades (usuários, grupos e funções) do IAM com base em sua atividade. Para fazer isso, visualize os eventos da sua conta no Event history (Histórico de eventos) do CloudTrail. Os logs de eventos do CloudTrail incluem informações de eventos detalhadas que você pode usar para alterar as permissões da política.

Um usuário ou perfil está tentando realizar uma ação na AWS e essa solicitação está sendo negada.

> Considere que o usuário ou o perfil tem permissão para realizar a ação. Se esse for o caso, você poderá adicionar a ação e até mesmo o ARN do recurso que eles tentou para acessar a sua política.

Um usuário ou um perfil tem permissões que não estão sendo usadas.

> Considere remover essas permissões da política deles. Certifique-se de que suas políticas concedem o menor privilégio necessário para executar apenas as ações necessárias.

Para obter mais informações sobre o uso do CloudTrail, consulteVisualizar eventos do CloudTrail no console do CloudTrail noGuia do usuário do AWS CloudTrail.

Solução de problemas de documentos de políticas JSON

Você pode diagnosticar e resolver problemas relacionados aos documentos de políticas JSON.

Validar suas políticas

Quando você cria ou edita uma política JSON, o IAM pode executar a validação de políticas para ajudar você a criar uma política eficaz. O IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de política do IAM. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer.

Não tenho permissões para validação de política no editor JSON

No Console de gerenciamento da AWS, você pode receber o seguinte erro se não tiver permissões para visualizar os resultados da validação da política do IAM Access Analyzer:

```
You need permissions. You do not have the permissions required to perform this operation. Ask your administrator to add permissions.
```

Para corrigir este erro, peça ao administrador para adicionar a permissão `access-analyzer:ValidatePolicy` para você.

Mais de um objeto de política JSON

Uma política do IAM deve consistir em um único objeto JSON. Você denota um objeto colocando chaves { } em torno. Você pode aninhar outros objetos dentro de um objeto JSON incorporando chaves adicionais { } dentro do par externo. Uma política deve conter somente um par externo de chaves { }. O exemplo a seguir é incorreto, pois contém dois objetos no nível superior (destacados em *vermelho*):

JSON

```
{
      "Version":"2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
    }
    {
      "Statement": {
         "Effect": "Allow",
         "Action": "s3:*",
         "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

No entanto, você pode atender à intenção do exemplo anterior com o uso da gramática correta da política. Em vez de incluir dois objetos completos da política, cada um com seu próprio elemento `Statement`, você pode combinar dois blocos em um único elemento `Statement`. O elemento `Statement` tem um conjunto de dois objetos como seu valor, como mostrado no seguinte exemplo (destacado em negrito):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
```

```
      ]
    }
```

Mais de um elemento de instrução JSON

À primeira vista, esse erro pode parecer uma variação da seção anterior. No entanto, sintaticamente é um tipo diferente de erro. No exemplo a seguir, há somente um objeto da política, conforme indicado por um único par de chaves { } no nível superior. No entanto, esse objeto contém dois elementos `Statement` dentro de si.

Uma política do IAM deve conter apenas um elemento `Statement`, consistindo no nome (`Statement`) que aparece à esquerda do sinal de dois pontos, seguido pelo valor à direita. O valor de um elemento `Statement` deve ser um objeto, denotado por chaves { }, contendo um elemento `Effect`, um elemento `Action` e um elemento `Resource`. O exemplo a seguir é incorreto, pois contém dois elementos `Statement` no objeto da política (destacados em *vermelho*):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Um objeto de valor pode ser um conjunto de vários objetos de valor. Para resolver esse problema, combine os dois elementos `Statement` em um elemento com um conjunto de objetos, conforme mostrado no seguinte exemplo (destacado em negrito):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
     ]
    }
```

O valor do elemento `Statement` é uma matriz de objetos. O conjunto nesse exemplo consiste em dois objetos, sendo que cada um deles por si só é um valor correto para um elemento `Statement`. Cada objeto na matriz é separado por vírgulas.

Mais de um efeito, ação ou elemento de recurso em um elemento de instrução JSON

No lado do valor do nome/valor `Statement`, o objeto deve conter apenas em um elemento `Effect`, um elemento `Action` e um elemento `Resource`. A política a seguir está incorreta, pois contém dois elementos `Effect` na `Statement`:

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

> **Note**
>
> O mecanismo de política não permite tais erros em políticas novas ou editadas. No entanto, o mecanismo de política continua a permitir a execução de políticas que foram salvas antes da atualização do mecanismo. O comportamento das políticas existentes com o erro é o seguinte:
>
> - Vários elementos `Effect`: apenas o último elemento `Effect` é observado. Os outros são ignorados.
> - Elementos `Action` múltiplos: todos os elementos `Action` são combinados internamente e tratados como se fossem uma lista única.
> - Elementos `Resource` múltiplos: todos os elementos `Resource` são combinados internamente e tratados como se fossem uma lista única.
>
> O mecanismo da política não permite salvar qualquer política com erros de sintaxe. Corrija os erros na política antes de salvá-los. Revise e corrija todas as recomendações de validação de políticas para suas políticas.

Em cada caso, a solução é remover o elemento extra incorreto. Para elementos `Effect`, é simples: se você deseja que o exemplo anterior negue permissões para instâncias do Amazon EC2, remova a linha `"Effect": "Allow",` da política da seguinte forma:

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:*",
        "Resource": "*"
      }
    }
```

No entanto, se o elemento duplicado é `Action` ou `Resource`, a resolução pode ser mais complicada. Você pode ter várias ações que deseja permitir (ou negar) ou você pode controlar

o acesso de vários recursos. Por exemplo, o exemplo a seguir está incorreto porque tem vários elementos `Resource` (destacados em *vermelho*):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
      }
    }
```

Cada um dos elementos necessários em um objeto de valor em um elemento `Statement` pode estar presente somente uma vez. A solução é colocar cada valor em um conjunto. O exemplo a seguir ilustra isso transformando dois elementos separados do recurso em um elemento `Resource` com uma matriz como o objeto de valor (destacado em negrito):

JSON

```
{
      "Version":"2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket",
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
      }
    }
```

Elemento de versão JSON ausente

Um elemento de política `Version` é diferente de uma versão de política. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. Em comparação, uma

versão da política é criada quando você altera uma política gerenciada pelo cliente no IAM. A política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. Para saber mais sobre o elemento de política `Version`, consulte Elementos de política JSON do IAM: Version. Para saber mais sobre as versões de política, consulte the section called "Versionamento de políticas do IAM".

À medida que os recursos da AWS evoluem, novos recursos são adicionados às políticas do IAM para oferecer suporte a esses recursos. Às vezes, uma atualização da sintaxe da política inclui um novo número de versão. Se você usar os recursos mais recentes da gramática da política na sua política, será necessário saber qual versão do mecanismo de análise de política você está usando.

A versão de política padrão é "2008-10-17." Se você deseja usar qualquer recurso de política que foi introduzido posteriormente, será necessário especificar o número da versão que dá suporte ao recurso desejado. Recomendamos que você sempre inclua o número da versão da sintaxe de política mais recente, que atualmente é `"Version": "2012-10-17"`.

Por exemplo, a política a seguir está incorreta, pois usa uma variável `${...}` de política no ARN para um recurso. Mas ela não especifica uma versão de sintaxe de política que ofereça suporte às variáveis de política (realçada em *vermelho*):

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Adicionar um elemento `Version` na parte superior da política com o valor `2012-10-17`, a primeira versão da API do IAM que oferece suporte às variáveis de política, resolve este problema (destacado em negrito):

JSON

```
{
  "Version":"2012-10-17",
  "Statement":
  {
```

```
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Solucionar problemas de chaves de acesso e chaves de segurança FIDO

Use estas informações para ajudá-lo a diagnosticar problemas comuns que você pode encontrar ao trabalhar com chaves de segurança FIDO2.

Tópicos

- Não consigo habilitar minha chave de segurança FIDO
- Não consigo fazer login usando minha chave de segurança FIDO
- Perdi ou quebrei minha chave de segurança FIDO
- Outros problemas

Não consigo habilitar minha chave de segurança FIDO

Consulte as soluções a seguir de acordo com seu status como um usuário ou administrador do sistema do IAM

Usuários do IAM

Se você não conseguir habilitar sua chave de segurança FIDO, verifique o seguinte:

- Você está usando uma configuração com suporte?

  O IAM é compatível com dispositivos de segurança FIDO2 que se conectam aos dispositivos por USB, Bluetooth ou NFC. O IAM também é compatível com os autenticadores de plataforma, como TouchID ou FaceID. O IAM não oferece suporte ao registro de chave de acesso local para o Windows Hello. Para criar e usar chaves de acesso, os usuários do Windows devem usar a autenticação entre dispositivos, na qual você usa uma chave de acesso de um dispositivo, como um dispositivo móvel, ou uma chave de segurança de hardware, para entrar em outro dispositivo, como um laptop.

Para obter informações sobre os dispositivos e navegadores que você pode usar com WebAuthn e AWS, consulte Configurações compatíveis com o uso de chaves de acesso e chaves de segurança.

- Você está usando algum plug-in de navegador?

  AWSA não é compatível com o uso de plug-ins para adicionar suporte de navegador a WebAuthn. Em vez disso, use um navegador que ofereça suporte nativo ao padrão WebAuthn.

  Mesmo que você esteja usando um navegador compatível, pode haver um plug-in que seja incompatível com WebAuthn. Um plug-in incompatível pode impedir você de habilitar e usar sua chave de segurança compatível com FIDO. Desabilite todos os plug-ins que possam ser incompatíveis e reinicie seu navegador. Em seguida, tente habilitar novamente a chave de segurança FIDO.

- Você tem as permissões apropriadas?

  Se você não tiver qualquer um dos problemas de compatibilidade acima, talvez não tenha as permissões apropriadas. Entre em contato com o administrador do sistema.

Administradores de sistemas

Caso seus usuários do IAM não consigam habilitar as chaves de segurança FIDO mesmo usando uma configuração compatível, verifique as permissões deles. Para obter um exemplo detalhado, consulte Tutorial do IAM: Permitir que os usuários gerenciem suas credenciais e configurações de MFA.

Não consigo fazer login usando minha chave de segurança FIDO

Se você não conseguir se conectar ao Console de gerenciamento da AWS usando a chave de segurança FIDO, primeiro consulte Configurações compatíveis com o uso de chaves de acesso e chaves de segurança. Se você estiver usando uma configuração com suporte mas não consegue fazer login, entre em contato com o administrador do sistema para obter assistência.

Perdi ou quebrei minha chave de segurança FIDO

Até oito dispositivos de MFA de qualquer combinação dos tipos de MFA atualmente compatíveis podem ser atribuídos a um usuário. Com vários dispositivos de MFA, basta um dispositivo de MFA para acessar o Console de gerenciamento da AWS. Substituir uma chave de segurança FIDO é

semelhante a substituir um token de hardware TOTP. Se você perder ou quebrar qualquer tipo de dispositivo de MFA, consulte Recuperar uma identidade protegida por MFA no IAM.

Outros problemas

Se você tiver um problema com chaves de segurança FIDO não tratado aqui, faça um os procedimentos a seguir:

- Usuários do IAM: entre em contato com o administrador do sistema.
- Usuários raiz da Conta da AWS: entre em contato com o AWS Support.

Solucionar problemas de perfis do IAM

Use estas informações para ajudar você a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com funções do IAM.

Tópicos

- Não consigo assumir uma função
- Uma nova função apareceu na minha conta da AWS
- Não consigo editar ou excluir um perfil na minha Conta da AWS
- Não estou autorizado a executar: iam:PassRole
- Por que não é possível assumir uma função com uma sessão de 12 horas? (AWS CLI, API da AWS)
- Recebo um erro quando tento alternar funções no console do IAM
- Minha função tem uma política que permite que eu execute uma ação, mas recebo a mensagem "access denied (acesso negado)"
- O serviço não criou a versão da política padrão da função
- Não há caso de uso para uma função de serviço no console

Não consigo assumir uma função

Verifique o seguinte:

- Para permitir que os usuários assumam novamente o perfil atual em uma sessão de perfil, especifique o ARN do perfil ou o ARN da Conta da AWS como entidade principal na política de confiança do perfil. Os Serviços da AWS que fornecem recursos computacionais, como

o Amazon EC2, Amazon ECS, Amazon EKS e Lambda, fornecem credenciais temporárias e atualizam automaticamente essas credenciais. Isso garante que você tenha sempre um conjunto de credenciais válido. Nesses serviços, não é necessário assumir novamente a função atual para obter credenciais temporárias. Porém, se pretender passar tags de sessão ou uma política de sessão, você precisará assumir novamente a função atual. Para saber como modificar uma política de confiança de função para adicionar o ARN da função de entidade principal ou o ARN da Conta da AWS, consulte Atualizar a política de confiança de um perfil.

- Ao assumir uma função usando o Console de gerenciamento da AWS, use o nome exato da sua função. Os nomes das funções diferenciam letras maiúsculas de minúsculas quando uma função é assumida.
- Ao assumir uma função usando a API do AWS STS ou a AWS CLI, use o nome exato da sua função no ARN. Os nomes das funções diferenciam letras maiúsculas de minúsculas quando uma função é assumida.
- Ao assumir um perfil usando um provedor de identidade federado baseado em SAML e com a criptografia SAML habilitada, certifique-se de ter carregado uma chave de descriptografia privada válida para o provedor de identidade SAML. Para obter mais informações, consulte Gerenciar chaves de criptografia SAML.
- Verifique se a política do IAM concede a você permissão para chamar `sts:AssumeRole` para a função que você deseja assumir. O elemento `Action` da política do IAM deve permitir chamar a ação `AssumeRole`. Além disso, o elemento `Resource` da política do IAM deve especificar a função que você deseja assumir. Por exemplo, o elemento `Resource` pode especificar uma função pelo Amazon Resource Name (ARN – Nome de recurso da Amazon) ou usando um curinga (*). Por exemplo, pelo menos uma política aplicável a você deve conceder permissões semelhantes às seguintes:

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Verifique se a identidade do IAM está etiquetada com alguma etiqueta exigida pela política do IAM. Por exemplo, nas permissões de política a seguir, o elemento `Condition` exige que você, como o principal que solicita assumir a função, tenha uma tag específica. Você deve estar marcado com `department = HR` ou `department = CS`. Do contrário, você não poderá pressupor a função. Para saber mais sobre como etiquetar usuários e funções do IAM, consulte the section called “Tags para recursos do IAM”.

```
    "Effect": "Allow",
```

```
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/department": [
            "HR",
            "CS"
        ]}}
```

- Verifique se você atende a todas as condições especificadas na política de confiança da função. Uma `Condition` pode especificar uma data de expiração, um ID externo ou que uma solicitação venha apenas de endereços IP específicos. Considere o seguinte exemplo: se a data atual for qualquer período após a data especificada, a política nunca será correspondida e não concederá a você a permissão para assumir a função.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
    "Condition": {
        "DateLessThan" : {
            "aws:CurrentTime" : "2016-05-01T12:00:00Z"
        }
    }
```

- Verifique se a Conta da AWS da qual você está chamando `AssumeRole` seja uma entidade confiável para o perfil que está assumindo. Entidades confiáveis são definidas como `Principal` em uma política de confiança da função. O exemplo a seguir é uma política de confiança anexada à função que você deseja assumir. Nesse exemplo, o ID da conta com o usuário do IAM com o qual você fez login deve ser 123456789012. Se o número da conta não estiver listado no elemento `Principal` da política de confiança da função, você não poderá assumir a função. Não importa quais permissões sejam concedidas a você nas políticas de acesso. Observe que a política de exemplo limita as permissões às ações que ocorrem entre 1° de julho de 2017 e 31 de dezembro de 2017 (UTC), inclusive. Se você fizer login antes ou depois dessas datas, a política não corresponderá e você não poderá assumir a função.

```
    "Effect": "Allow",
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
```

- Identidade da origem: os administradores podem configurar funções para exigir que identidades passem uma string personalizada que identifique a pessoa ou a aplicação que está executando ações na AWS, chamada Identidade de origem. Verifique se a função que está sendo assumida requer que uma identidade de origem esteja definida. Para obter mais informações sobre identidade de origem, consulte Monitorar e controlar ações realizadas com funções assumidas.

Uma nova função apareceu na minha conta da AWS

Alguns serviços da AWS requerem que você use um tipo exclusivo de função de serviço que seja diretamente vinculada ao serviço. Essa função vinculada ao serviço é predefinida pelo serviço e inclui todas as permissões que o serviço requer. Isso facilita a configuração de um serviço, pois você não precisa adicionar manualmente as permissões necessárias. Para obter informações gerais sobre funções vinculadas ao serviço, consulte Criar um perfil vinculado ao serviço.

Você já pode estar usando um serviço quando ele começa a oferecer suporte a funções vinculadas ao serviço. Se esse for o caso, você pode receber um e-mail informando sobre uma nova função na sua conta. Essa função inclui todas as permissões de que o serviço precisa para executar ações em seu nome. Você não precisa realizar nenhuma ação para oferecer suporte a essa função. No entanto, você não deve excluir a função de sua conta. Isso pode remover permissões de que o serviço precisa para acessar recursos da AWS. Você pode visualizar as funções vinculadas ao serviço na sua conta acessando a página Roles (Funções) do console do IAM. As funções vinculadas ao serviço aparecem com (Função vinculada ao serviço) na coluna Entidades confiáveis da tabela.

Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Para obter informações sobre como usar a função vinculada a um serviço, escolha o link Sim.

Não consigo editar ou excluir um perfil na minha Conta da AWS

Você não pode excluir ou editar as permissões de uma função vinculada ao serviço no IAM. Essas funções incluem confianças e permissões predefinidas exigidas pelo serviço para executar ações em seu nome. Você pode usar o console, a AWS CLI ou a API do IAM para editar somente a descrição de uma função vinculada ao serviço. Você pode visualizar as funções vinculadas ao serviço na sua conta acessando a página Roles (Funções) do IAM no console. As funções vinculadas ao serviço aparecem com (Função vinculada ao serviço) na coluna Entidades confiáveis da tabela. Um banner na página Resumo da função também indica que a função é uma função vinculada ao serviço. Você

pode gerenciar e excluir essas funções apenas por meio do serviço vinculado, caso o serviço ofereça suporte para a ação. Tenha cuidado ao modificar ou excluir uma função vinculada ao serviço, pois isso pode remover permissões de que o serviço precisa para acessar recursos da AWS.

Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço.

Não estou autorizado a executar: iam:PassRole

Quando cria uma função vinculada ao serviço, você precisa ter permissão para passar essa função para o serviço. Alguns serviços criam automaticamente uma função vinculada ao serviço na sua conta quando você executa uma ação nesse serviço. Por exemplo, o Amazon EC2 Auto Scaling cria a função vinculada ao serviço `AWSServiceRoleForAutoScaling` para você quando você cria um grupo do Auto Scaling pela primeira vez. Se você tentar criar um grupo de Auto Scaling sem a permissão `PassRole`, receberá o seguinte erro:

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Para corrigir este erro, peça ao administrador para adicionar a permissão `iam:PassRole` para você.

Para saber quais serviços dão suporte a funções vinculadas ao serviço, consulte AWSServiços da que funcionam com o IAM. Para saber se um serviço cria automaticamente uma função vinculada ao serviço para você, escolha o link Sim para visualizar a documentação da função vinculada ao serviço.

Por que não é possível assumir uma função com uma sessão de 12 horas? (AWS CLI, API da AWS)

Quando usa as operações de API AWS STS ou as operações da CLI `AssumeRole*` do `assume-role*` para assumir uma função, você pode especificar um valor para o parâmetro `DurationSeconds`. É possível especificar um valor de 900 segundos (15 minutos) até a configuração da Duração máxima da sessão para a função. Se você especificar um valor maior do que o configurado, a operação falhará. Essa configuração pode ter um valor máximo de 12 horas. Por exemplo, se você especificar uma duração de 12 horas para a sessão, mas o administrador

definir a duração máxima da sessão como 6 horas, a operação falhará. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil.

Se você usar o encadeamento de funções (usar uma função para assumir uma segunda função), sua sessão será limitada a um máximo de uma hora. Se você usar o parâmetro `DurationSeconds` para fornecer um valor maior do que uma hora, a operação falhará.

Recebo um erro quando tento alternar funções no console do IAM

As informações inseridas na página Trocar de função devem corresponder às informações da função. Caso contrário, a operação falha e você recebe o seguinte erro:

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Se você receber esse erro, confirme se as seguintes informações estão corretas:

- ID ou alias da conta: o ID da Conta da AWS é um número de 12 dígitos. Sua conta pode ter um alias, que é um identificador fácil, como o nome da empresa, que pode ser usado em vez do ID da Conta da AWS. É possível usar o ID ou o alias da conta nesse campo.
- Nome da função: os nomes das funções diferenciam letras maiúsculas de minúsculas. O ID da conta e o nome da função devem corresponder ao que está configurado para a função.

Se continuar a receber uma mensagem de erro, entre em contato com o administrador para verificar as informações anteriores. A política de confiança de função ou a política de usuário do IAM pode limitar o acesso. O administrador pode verificar as permissões para essas políticas.

Minha função tem uma política que permite que eu execute uma ação, mas recebo a mensagem "access denied (acesso negado)"

Sua sessão de função pode estar limitada por políticas de sessão. Quando solicita credenciais de segurança temporárias de forma programática usando o AWS STS, você pode opcionalmente passar políticas de sessão em linha ou gerenciadas. As políticas de sessão são políticas avançadas que você passa como um parâmetro ao criar uma sessão de credenciais temporárias de forma programática para uma função. É possível passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. É possível usar o parâmetro `PolicyArns` para especificar até 10 políticas de sessão gerenciadas. As permissões da sessão resultante são a interseção das políticas baseadas em identidade da função e das políticas de sessão. Como alternativa, se o

administrador ou um programa personalizado fornecer credenciais temporárias a você, ele poderá incluir uma política de sessão para limitar seu acesso.

O serviço não criou a versão da política padrão da função

Uma função de serviço é uma função que um serviço assume para realizar ações em seu nome na sua conta. Ao configurar alguns ambientes de serviço da AWS, você deve definir uma função a ser assumida pelo serviço. Em alguns casos, o serviço cria a função de serviço e sua política no IAM para você. Embora você possa modificar ou excluir a função de serviço e sua política no IAM, a AWS não recomenda fazer isso. A função e a política devem ser utilizadas apenas por esse serviço. Se você editar a política e configurar outro ambiente, quando o serviço tentar usar a mesma função e política, a operação poderá falhar.

Por exemplo, quando você usa o AWS CodeBuild pela primeira vez, o serviço cria uma função chamada `codebuild-RWBCore-service-role`. Essa função de serviço usa a política chamada `codebuild-RWBCore-managed-policy`. Se você editar a política, ela criará outra versão e salvará essa versão como padrão. Se você executar uma operação subsequente no AWS CodeBuild, o serviço poderá tentar atualizar a política. Se isso acontecer, você receberá o seguinte erro:

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
RWBCore-service-role role. To continue, detach the policy from any other
identities and then delete the policy and the role.
```

Se você receber esse erro, faça alterações no IAM antes de continuar com a operação do serviço. Primeiro, defina a versão de política padrão como V1 e tente executar a operação novamente. Se V1 tiver sido excluída anteriormente ou se escolher V1 não funcionar, limpe e exclua a política e a função existentes.

Para obter mais informações sobre como editar políticas gerenciadas, consulte Edição de políticas gerenciadas pelo cliente (console). Para obter mais informações sobre as versões de política, consulte Versionamento de políticas do IAM.

Como excluir uma função de serviço e sua política

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM, em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas.

3. Na lista de políticas, escolha o nome da política que deseja excluir.
4. Escolha a guia Entidades anexadas para visualizar quais usuários, grupos ou perfis do IAM usam essa política. Se qualquer uma dessas identidades usar a política, conclua as seguintes tarefas:
    a. Crie uma política gerenciada com as permissões necessárias. Para garantir que as identidades tenham as mesmas permissões antes e depois de suas ações, copie o documento de política JSON da política existente. Depois, crie a nova política gerenciada e cole o documento JSON como descrito em Criar políticas usando o editor de JSON.
    b. Para cada identidade afetada, anexe a nova política e desanexe a antiga. Para obter mais informações, consulte Adicionar e remover permissões de identidade do IAM.
5. No painel de navegação, escolha Perfis.
6. Na lista de funções, escolha o nome da função que deseja excluir.
7. Selecione a guia Relações de confiança para visualizar quais entidades podem assumir a função. Se qualquer entidade diferente do serviço estiver listada, conclua as seguintes tarefas:
    a. Crie uma função que confie nessas entidades.
    b. A política criada na etapa anterior. Se você pulou essa etapa, crie a política gerenciada agora.
    c. Notifique as pessoas que estavam assumindo a função de que elas não podem mais fazer isso. Forneça informações sobre como assumir a nova função e ter as mesmas permissões.
8. Exclua a política.
9. Exclua o perfil.

Não há caso de uso para uma função de serviço no console

Alguns serviços exigem que você crie manualmente uma função de serviço para conceder permissões de serviço para executar ações em seu nome. Se o serviço não estiver listado no console do IAM, você deverá listar manualmente o serviço como a entidade confiável. Se a documentação do serviço ou recurso que você estiver usando não incluir instruções para listar o serviço como o principal confiável, forneça feedback para a página.

Para criar uma função de serviço manualmente, você precisa saber o principal do serviço para o serviço que assumirá a função. Um principal do serviço é um identificador que é usado para conceder permissões a um serviço. O escopo principal do serviço é definido pelo serviço.

Você pode encontrar o principal de serviço para alguns serviços, verificando o seguinte:

1. Abra o AWSServiços da que funcionam com o IAM.
2. Verifique se o serviço mostra Sim na coluna Service-linked roles (Funções vinculadas ao serviço).
3. Escolha o link Sim para visualizar a documentação do perfil vinculado a serviço desse serviço.
4. Localize a seção Service-Linked Role Permissions (Permissões da função vinculada ao serviço) desse serviço para visualizar o principal do serviço.

Criar uma função de serviço manualmente usando AWS CLI comandos ou operações deAWS API. Para criar manualmente uma função de serviço usando o console do IAM, conclua as seguintes tarefas:

1. Crie uma função do IAM usando o ID da conta. Não associe uma política nem conceda nenhuma permissão. Para obter detalhes, consulte Criar um perfil para conceder permissões a um usuário do IAM.
2. Abra a função e edite o relacionamento de confiança. Em vez de confiar na conta, a função deve confiar no serviço. Por exemplo, atualize o seguinte elemento `Principal`:

   ```
   "Principal": { "AWS": "arn:aws:iam::123456789012:root" }
   ```

   Altere a entidade para o valor do seu serviço, como IAM.

   ```
   "Principal": { "Service": "iam.amazonaws.com" }
   ```
3. Adicione as permissões que o serviço exige associando políticas de permissões à função.
4. Retorne ao serviço que exige as permissões e use o método documentado para notificar o serviço sobre a nova função de serviço.

Solucionar problemas do IAM e Amazon EC2

As informações a seguir podem ajudar você a solucionar problemas do IAM com o Amazon EC2.

Tópicos

- Ao tentar iniciar uma instância, não vejo o perfil na lista de perfis do IAM do console do Amazon EC2
- As credenciais na minha instância são da função incorreta

- Quando tento chamar AddRoleToInstanceProfile, recebo um erro AccessDenied.
- Amazon EC2: quando tento iniciar uma instância com um perfil, recebo um erro AccessDenied
- Não é possível acessar as credenciais de segurança temporárias em minha instância do EC2.
- O que os erros do documento info na subárvore do IAM significam?

Ao tentar iniciar uma instância, não vejo o perfil na lista de perfis do IAM do console do Amazon EC2

Verifique o seguinte:

- Se você estiver conectado a um usuário do IAM, verifique se tem permissão para chamar `ListInstanceProfiles`. Para obter mais informações sobre as permissões necessárias para trabalhar com perfis, consulte Permissões necessárias para usar funções com o Amazon EC2. Para obter informações sobre como adicionar permissões a um usuário, consulte Gerenciar políticas do IAM.

  Se você não puder modificar suas próprias permissões, deverá entrar em contato com um administrador que possa trabalhar com o IAM para atualizar suas permissões.
- Se você criou um perfil usando a CLI ou a API do IAM, verifique as seguintes condições:
  - Você criou um perfil de instância e adicionou o perfil a ele.
  - Você usou o mesmo nome para o perfil e o perfil de instância. Se você nomear seu perfil e o perfil de instância de forma diferente, não verá o nome correto do perfil no console do Amazon EC2.

  A lista IAM Role (Função do IAM) no console do Amazon EC2 lista os nomes dos perfis de instância, não os nomes das funções. Você precisará selecionar o nome do perfil de instância que contém a função desejada. Para obter detalhes sobre os perfis de instância, consulte Usar perfis de instância.

  > **Note**
  >
  > Se você usar o console do IAM para criar funções, não precisará trabalhar com perfis de instância. Para cada função criada no console do IAM, um perfil de instância é criado com o mesmo nome da função, e a função é automaticamente adicionada a esse perfil. Um

perfil de instância pode conter somente uma função do IAM e esse limite não pode ser aumentado.

As credenciais na minha instância são da função incorreta

A função no perfil da instância pode ter sido substituída recentemente. Nesse caso, seu aplicativo precisará aguardar a próxima rotação de credenciais programada automaticamente para que as credenciais de sua função fiquem disponíveis.

Para forçar a alteração, desassocie o perfil de instância, associe o perfil de instância, ou interrompa a instância e, em seguida, reinicie-a.

Quando tento chamar **`AddRoleToInstanceProfile`**, recebo um erro **`AccessDenied`**.

Se você estiver fazendo solicitações como um usuário IAM, verifique se tem as seguintes permissões:

- `iam:AddRoleToInstanceProfile` com o recurso correspondente ao nome de região da Amazon (ARN) do perfil da instância (por exemplo, `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

Para obter mais informações sobre as permissões necessárias para trabalhar com perfis, consulte Como faço para começar?. Para obter informações sobre como adicionar permissões a um usuário, consulte Gerenciar políticas do IAM.

Amazon EC2: quando tento iniciar uma instância com um perfil, recebo um erro **`AccessDenied`**

Verifique o seguinte:

- Execute uma instância sem um perfil de instância. Isso ajudará a garantir que o problema seja limitado às funções do IAM para instâncias do Amazon EC2.
- Se você estiver fazendo solicitações como um usuário IAM, verifique se tem as seguintes permissões:
  - `ec2:RunInstances` com um caractere curinga ("*")

  - `iam:PassRole` com o recurso correspondente ao nome de região da Amazon (ARN) da função (por exemplo, `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Chame a ação `GetInstanceProfile` do IAM para garantir que você esteja usando um nome de perfil de instância válido ou um ARN de perfil de instância válido. Para obter mais informações, consulte Usar funções IAM com instâncias do Amazon EC2.
- Chame a ação `GetInstanceProfile` do IAM para garantir que o perfil da instância tenha uma função. Ocorrerá falha em perfis de instância vazias com o erro `AccessDenied`. Para obter mais informações sobre a criação de uma função, consulte Criar um perfil do IAM.

Para obter mais informações sobre as permissões necessárias para trabalhar com perfis, consulte Como faço para começar?. Para obter informações sobre como adicionar permissões a um usuário, consulte Gerenciar políticas do IAM.

Não é possível acessar as credenciais de segurança temporárias em minha instância do EC2.

Para acessar credenciais de segurança temporárias em sua instância do EC2, você deve primeiro usar o console do IAM para criar uma função. Depois, execute uma instância do EC2 que usa essa função e examine a instância em execução. Para obter mais informações, consulte How Do I Get Started? (Como começar?) em Use um perfil do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2.

Se você ainda não conseguir acessar as credenciais de segurança temporárias na instância do EC2, verifique o seguinte:

- É possível acessar outra parte do serviço de metadados da instância (IMDS)? Se não, verifique se não há regras de firewall bloqueando o acesso a solicitações para o IMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- A subárvore `iam` do IMDS existe? Caso não exista, verifique se sua instância tem um perfil de instância do IAM associado a ela chamando a operação de API `DescribeInstances` do EC2 ou usando o comando `aws ec2 describe-instances` da CLI.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Verifique se há erro no documento `info` na subárvore do IAM. Se houver erro, consulte O que os erros do documento `info` na subárvore do IAM significam? para obter mais informações.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

O que os erros do documento **`info`** na subárvore do IAM significam?

O documento **`iam/info`** indica **`"Code":"InstanceProfileNotFound"`**

Seu perfil de instância do IAM foi excluído e o Amazon EC2 não pode mais fornecer credenciais para sua instância. Você deve anexar um perfil de instância válido à sua instância do Amazon EC2.

Se houver um perfil de instância com esse nome, verifique se o perfil não foi excluído e outro foi criado com o mesmo nome:

1. Chame a operação `GetInstanceProfile` do IAM para obter o `InstanceProfileId`.
2. Chame a operação `DescribeInstances` do Amazon EC2 para obter o `IamInstanceProfileId` para a instância.
3. Verifique se o `InstanceProfileId` da operação do IAM corresponde ao `IamInstanceProfileId` da operação do Amazon EC2.

Se os IDs forem diferentes, o perfil de instância anexado às suas instâncias não será mais válido. Você deve anexar um perfil de instância válido à instância.

O documento **`iam/info`** indica um sucesso, mas indica **`"Message":"Instance Profile does not contain a role..."`**

A função foi removida do perfil da instância pela ação `RemoveRoleFromInstanceProfile` do IAM. Você pode usar a ação `AddRoleToInstanceProfile` do IAM para anexar uma função ao perfil da instância. O aplicativo precisará aguardar até que a próxima atualização programada acesse as credenciais para a função.

Para forçar a alteração, desassocie o perfil de instância, associe o perfil de instância, ou interrompa a instância e, em seguida, reinicie-a.

O documento **`iam/security-credentials/[role-name]`** indica **`"Code":"AssumeRoleUnauthorizedAccess"`**

O Amazon EC2 não tem permissão para assumir a função. A permissão para assumir a função é controlada pela política de confiança anexada à função, como o exemplo a seguir. Use a API `UpdateAssumeRolePolicy` do IAM para atualizar a política de confiança.

JSON

```
{"Version":"2012-10-17",       "Statement": [{"Effect": "Allow","Principal":
 {"Service": ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

O aplicativo precisará aguardar até que a próxima atualização programada automaticamente acesse as credenciais para a função.

Para forçar a alteração, desassocie o perfil de instância, associe o perfil de instância, ou interrompa a instância e, em seguida, reinicie-a.

Solucionar problemas do IAM e Amazon S3

Use estas informações para ajudar você a solucionar e corrigir problemas que você pode encontrar ao trabalhar com o Amazon S3 e o IAM.

Como faço para conceder acesso anônimo a um bucket do Amazon S3?

Você pode usar uma política de bucket do Amazon S3 que especifique um caractere curinga (*) no elemento `principal`, o que significa que qualquer pessoa pode acessar o bucket. Com o acesso anônimo, qualquer pessoa (incluindo usuários sem uma Conta da AWS), poderá acessar o bucket. Para obter uma amostra de política, consulte Casos de exemplo para políticas de bucket do Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

Estou conectado como um usuário-raiz da Conta da AWS. Por que não consigo acessar um bucket do Amazon S3 com minha conta?

Em alguns casos, você pode ter um usuário do IAM com acesso total ao IAM e ao Amazon S3. Se o usuário do IAM atribuir uma política de bucket a um bucket do Amazon S3 e não especificar o usuário raiz da como uma entidade de segurança, o usuário raiz terá o acesso negado a esse

bucket. No entanto, como o usuário raiz, você ainda pode acessar o bucket. Para fazer isso, modifique a política de bucket para permitir o acesso do usuário raiz no console do Amazon S3 ou da AWS CLI. Use a seguinte entidade principal, substituindo *123456789012* pelo ID da Conta da AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Solucionar problemas de federação do SAML com o IAM

Use estas informações para ajudá-lo a diagnosticar e corrigir problemas que encontrar ao trabalhar com o SAML 2.0 e a federação com o AWS Identity and Access Management.

Tópicos

- Erro: sua solicitação incluiu uma resposta SAML inválida. Para sair, clique aqui.
- Erro: RoleSessionName é necessário em AuthnResponse (serviço: AWSSecurityTokenService; código de status: 400; código de erro: InvalidIdentityToken)
- Erro: sem autorização para executar sts:AssumeRoleWithSAML (serviço: AWSSecurityTokenService; código de status: 403; código de erro: AccessDenied)
- Erro: RoleSessionName em AuthnResponse deve corresponder a [a-zA-Z_0-9+=,.@-]{2,64} (serviço: AWSSecurityTokenService; código de status: 400; código de erro: InvalidIdentityToken)
- Erro: A identidade-fonte deve corresponder a [a-zA-Z_0-9+=,.@-]{2,256} e não começar com "aws:" (serviço: AWSSecurityTokenService; código do status: 400; código do erro: InvalidIdentityToken)
- Erro: assinatura inválida da resposta (serviço: AWSSecurityTokenService; código de status 400; código de erro: InvalidIdentityToken)
- Erro: chave privada inválida.
- Erro: falha ao remover chave privada.
- Erro: falha ao remover a chave privada porque a ID da chave não corresponde a uma chave privada.
- Erro: falha ao assumir a função: emissor ausente no provedor especificado (serviço: AWSOpenIdDiscoveryService; código de status: 400; código de erro: AuthSamlInvalidSamlResponseException)
- Erro: não foi possível analisar os metadados.
- Erro: não é possível atualizar o provedor de identidade. Não há nenhuma atualização definida para metadados ou declaração de criptografia.

- Erro: não é possível definir o modo de criptografia de declaração como Obrigatório porque nenhuma chave privada foi fornecida.
- Erro: não é possível adicionar e remover chaves privadas na mesma solicitação. Defina um valor apenas para um dos dois parâmetros.
- Erro: o provedor especificado não existe.
- Erro: o valor de DurationSeconds solicitado excede o valor de MaxSessionDuration definido para esta função.
- Erro: limite máximo de 2 chaves privadas atingido.
- Erro: a resposta não contém o público necessário.

Erro: sua solicitação incluiu uma resposta SAML inválida. Para sair, clique aqui.

Esse erro pode ocorrer quando a resposta do SAML do provedor de identidade não incluir um atributo com o `Name` definido como `https://aws.amazon.com/SAML/Attributes/Role`. O atributo deve conter um ou mais elementos `AttributeValue`, cada um contendo um par de strings separado por vírgulas:

- O ARN de uma função para a qual o usuário pode ser mapeado
- O ARN do provedor SAML

O erro também pode ocorrer quando os valores do atributo SAML enviados pelo provedor de identidade têm um espaço em branco à esquerda ou à direita ou outros caracteres inválidos nos valores do atributo SAML. Para obter mais informações sobre os valores esperados para os atributos SAML, consulte Configurar declarações SAML para a resposta de autenticação

Para obter mais informações, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Visualizar uma resposta do SAML no navegador.

Erro: RoleSessionName é necessário em AuthnResponse (serviço: AWSSecurityTokenService; código de status: 400; código de erro: InvalidIdentityToken)

Esse erro pode ocorrer quando a resposta do SAML do provedor de identidade não incluir um atributo com o `Name` definido como `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. O valor do atributo é um identificador para o usuário e, geralmente, é um ID do usuário ou um endereço de e-mail.

Para obter mais informações, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Visualizar uma resposta do SAML no navegador.

Erro: sem autorização para executar sts:AssumeRoleWithSAML (serviço: AWSSecurityTokenService; código de status: 403; código de erro: AccessDenied)

Esse erro pode ocorrer se a função do IAM especificada na resposta do SAML estiver digitada incorretamente ou não existir. Use o nome exato da sua função, porque os nomes de função diferenciam maiúsculas de minúsculas. Corrija o nome da função na configuração do provedor de serviços do SAML.

Você terá o acesso permitido somente se a política de confiança da função incluir a ação `sts:AssumeRoleWithSAML`. Se a sua declaração SAML for configurada para usar o atributo `PrincipalTag`, a política de confiança também deverá incluir a ação `sts:TagSession`. Para obter mais informações sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

Esse erro pode ocorrer se você não tiver permissões `sts:SetSourceIdentity` em sua política de confiança de função. Se a sua declaração SAML for configurada para usar o atributo `SourceIdentity`, a política de confiança também deverá incluir a ação `sts:SetSourceIdentity`. Para obter mais informações sobre identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

Esse erro também poderá ocorrer se uma entidade principal federada não tiver permissões para assumir o perfil. A função deve ter uma política de confiança que especifique o ARN do provedor de identidade SAML do IAM como `Principal`. A função também contém as condições que controlam quais usuários podem assumir a função. Certifique-se de que os usuários atendam aos requisitos das condições.

Esse erro também pode ocorrer se a resposta do SAML não incluir um `Subject` contendo um `NameID`.

Para obter mais informações, consulte Habilitar o acesso das entidades principais federadas do SAML 2.0 ao Console de gerenciamento da AWS e Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Visualizar uma resposta do SAML no navegador.

Erro: RoleSessionName em AuthnResponse deve corresponder a [a-zA-Z_0-9+=,.@-]{2,64} (serviço: AWSSecurityTokenService; código de status: 400; código de erro: InvalidIdentityToken)

Esse erro pode ocorrer se o valor do atributo `RoleSessionName` for muito longo ou contiver caracteres inválidos. O comprimento máximo válido é de 64 caracteres.

Para obter mais informações, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Visualizar uma resposta do SAML no navegador.

Erro: A identidade-fonte deve corresponder a [a-zA-Z_0-9+=,.@-]{2,256} e não começar com **`"aws:"`** (serviço: AWSSecurityTokenService; código do status: 400; código do erro: InvalidIdentityToken)

Esse erro pode ocorrer se o valor do atributo `sourceIdentity` for muito longo ou contiver caracteres inválidos. O comprimento máximo válido é de 256 caracteres. Para obter mais informações sobre identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

Para obter mais informações sobre como criar declarações SAML, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Visualizar uma resposta do SAML no navegador.

Erro: assinatura inválida da resposta (serviço: AWSSecurityTokenService; código de status 400; código de erro: InvalidIdentityToken)

Esse erro poderá ocorrer quando os metadados de federação do provedor de identidade não corresponderem aos metadados do provedor de identidade do IAM. Por exemplo, o arquivo de

metadados para o provedor de serviços de identidade pode ter sido alterado para atualizar um certificado expirado. Faça download do arquivo de metadados SAML atualizado a partir de seu provedor de serviços de identidade. Atualize-o na entidade do provedor de identidade da AWS que você define no IAM com o comando da CLI entre plataformas `aws iam update-saml-provider` ou o cmdlet `Update-IAMSAMLProvider` do PowerShell.

Erro: chave privada inválida.

Esse erro pode ocorrer quando você não formata adequadamente seu arquivo de chave privada. Esse erro pode fornecer detalhes adicionais sobre o motivo da invalidade da chave privada:

- A chave está criptografada.
- O formato da chave não é reconhecido. O arquivo de chave privada deve ser um arquivo .pem.

Ao Criar um provedor de identidades SAML no IAM no Console de gerenciamento da AWS, você deve baixar a chave privada do seu provedor de identidade para fornecer ao IAM a fim de habilitar a criptografia. A chave privada deve ser um arquivo .pem que use o algoritmo de criptografia AES-GCM ou AES-CBC para descriptografar declarações SAML.

Erro: falha ao remover chave privada.

Esse erro pode ocorrer quando a criptografia SAML estiver definida como Obrigatória e sua solicitação possa remover a única chave de descriptografia privada do provedor de SAML do IAM. Para obter mais informações sobre a alternância de chaves privadas, consulte Gerenciar chaves de criptografia SAML.

Erro: falha ao remover a chave privada porque a ID da chave não corresponde a uma chave privada.

Esse erro pode ocorrer se o valor `keyId` da chave privada não corresponder a nenhum dos IDs de chave dos arquivos de chave privada do provedor de identidade.

Quando você usa as operações de API update-saml-provider ou UpdateSAMLProvider para remover chaves privadas de criptografia SAML, o valor em `RemovePrivateKey` deve ser um ID de chave válido para uma chave privada anexada ao seu provedor de identidade.

Erro: falha ao assumir a função: emissor ausente no provedor especificado (serviço: AWSOpenIdDiscoveryService; código de status: 400; código de erro: AuthSamlInvalidSamlResponseException)

Esse erro poderá ocorrer se o emissor na resposta do SAML não corresponder ao emissor declarado no arquivo de metadados da federação. O arquivo de metadados foi carregado na AWS quando você criou o provedor de identidade no IAM.

Erro: não foi possível analisar os metadados.

Este erro poderá ocorrer se você não formatar adequadamente seu arquivo de metadados.

Quando você criar ou gerenciar um provedor de identidade do SAML no Console de gerenciamento da AWS, deverá recuperar o documento de metadados do SAML do seu provedor de identidade.

Esse arquivo de metadados inclui o nome do emissor, informações de validade e chaves que podem ser usadas para validar a resposta de autenticação do SAML (declarações) que são recebidas do IdP. O arquivo de metadados deve ser codificado no formato UTF-8 sem a marca de ordem de bytes (BOM). Para remover a BOM, você pode codificar o arquivo como UTF-8 usando uma ferramenta de edição de texto, como o Notepad++.

O certificado x.509 incluído como parte do documento de metadados do SAML deve usar um tamanho de chave de, pelo menos, 1.024 bits. Além disso, o certificado x.509 também deve estar livre de extensões repetidas. É possível usar extensões, mas elas só podem aparecer uma vez no certificado. Se o certificado x.509 não atender a nenhuma das condições, a criação do IdP vai falhar e retornar um erro “Unable to parse metadata”.

Conforme definido pela Versão 1.0 do Perfil de Interoperabilidade de Metadados do SAML V2.0, o IAM não avalia nem toma medidas em relação à expiração dos certificados X.509 em documentos de metadados do SAML. Caso não esteja preocupado com certificados X.509 expirados, recomendamos monitorar as datas de expiração dos certificados e alterná-los de acordo com as políticas de governança e segurança da sua organização.

Erro: não é possível atualizar o provedor de identidade. Não há nenhuma atualização definida para metadados ou declaração de criptografia.

Esse erro pode ocorrer se você usar as operações `update-saml-provider` da CLI ou `UpdateSAMLProvider` da API, mas não fornecer valores de atualização nos parâmetros da

solicitação. Para obter mais informações sobre como atualizar seu provedor de SAML do IAM, consulte Criar um provedor de identidades SAML no IAM.

Erro: não é possível definir o modo de criptografia de declaração como Obrigatório porque nenhuma chave privada foi fornecida.

Esse erro pode ocorrer quando você ainda não tiver feito upload de uma chave de decodificação privada e tentar definir a criptografia SAML como Obrigatória sem incluir uma chave privada em sua solicitação.

Certifique-se de que haja uma chave privada definida para seu provedor de SAML do IAM ao usar as operações `create-saml-provider` da CLI, `CreateSAMLProvider` da API, `update-saml-provider` da CLI `UpdateSAMLProvider` ou da API para exigir declarações de SAML criptografadas.

Erro: não é possível adicionar e remover chaves privadas na mesma solicitação. Defina um valor apenas para um dos dois parâmetros.

Esse erro pode ocorrer quando os valores de adição e remoção de chaves privadas estão incluídos na mesma solicitação.

Ao usar as operações de API update-saml-provider ou UpdateSAMLProvider para alternar arquivos de chave privada de criptografia SAML, você só pode adicionar ou remover uma chave privada na sua solicitação. Se você adicionar uma chave privada ao remover uma chave privada, a operação falhará. Para obter mais informações sobre a alternância de chaves privadas, consulte Gerenciar chaves de criptografia SAML.

Erro: o provedor especificado não existe.

Esse erro pode ocorrer se o nome do provedor na asserção do SAML não corresponder ao nome do provedor no IAM. Para obter mais informações sobre como visualizar o nome do provedor, consulte Criar um provedor de identidades SAML no IAM.

Erro: o valor de DurationSeconds solicitado excede o valor de MaxSessionDuration definido para esta função.

Esse erro pode ocorrer se você assumir uma função a partir da AWS CLI ou da API.

Quando você usa as operações da CLI assume-role-with-saml ou da API AssumeRoleWithSAML para assumir uma função, pode especificar um valor para o parâmetro `DurationSeconds`. Você pode especificar um valor de 900 segundos (15 minutos) até o valor configurado de duração máxima da sessão para a função. Se você especificar um valor maior do que o configurado, a operação falhará. Por exemplo, se você especificar uma duração de 12 horas para a sessão, mas o administrador definir a duração máxima da sessão como 6 horas, a operação falhará. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil.

Erro: limite máximo de 2 chaves privadas atingido.

Esse erro pode ocorrer quando você tenta adicionar uma chave privada ao seu provedor de identidade.

É possível salvar até duas chaves privadas para cada provedor de identidade. Quando você usa as operações de API update-saml-provider ou UpdateSAMLProvider para adicionar uma terceira chave privada, a operação falha.

Antes de adicionar uma nova chave privada, remova as chaves privadas que expiraram. Para obter mais informações sobre a alternância de chaves privadas, consulte Gerenciar chaves de criptografia SAML.

Erro: a resposta não contém o público necessário.

Esse erro pode ocorrer se o URL do público não corresponder ao provedor de identidades na configuração de SAML. Certifique-se de que o identificador de parte confiável do provedor de identidades (IdP) corresponda exatamente ao URL do público (ID da entidade) fornecido na configuração de SAML.

Como o IAM funciona com outros serviços da AWS

Embora o IAM seja o principal serviço da AWS que você usará para gerenciar recursos do IAM, todos os outros serviços da AWS funcionam com o IAM para controlar o acesso aos recursos em sua conta.

- CloudFormation

  O CloudFormation se integra ao IAM permitindo que você defina e gerencie recursos do IAM como parte de seus modelos do CloudFormation. É possível pode usar o CloudFormation para especificar as permissões necessárias do IAM para outros recursos da AWS que você provisiona. O CloudFormation também oferece suporte ao uso de perfis do IAM para gerenciar as credenciais necessárias para provisionar e gerenciar sua infraestrutura da AWS, e seu recurso de detecção de desvios ajuda a manter a integridade de suas configurações do IAM.
- AWS CloudShell

  Quando você acessa o AWS CloudShell, sua autenticação e autorização são tratadas por meio do IAM. O AWS CloudShell é executado dentro do contexto de um perfil do IAM atribuído ao seu usuário ou conta. Quando você inicia o AWS CloudShell, ele gera automaticamente credenciais de segurança temporárias com base no perfil do IAM atribuído a você.
- SDKs da AWS

  Os SDKs da AWS funcionam com o IAM gerenciando o processo de autenticação e autorização, gerenciando credenciais da AWS e respeitando as permissões e políticas definidas no IAM para garantir que sua aplicação só possa acessar os recursos que está autorizada a usar. Os SDKs fornecem mecanismos para obter e usar credenciais de segurança temporárias, além de validar as permissões necessárias para as operações de sua aplicação.

Para obter uma lista dos serviços da AWS que funcionam com o IAM e dos recursos do IAM compatíveis com os serviços, consulte AWSServiços da que funcionam com o IAM.

Criar recursos do IAM com o CloudFormation

AWS Identity and Access Management está integrado a AWS CloudFormation, um serviço que ajuda a modelar e configurar seus atributos AWS para que você possa gastar menos tempo criando e gerenciando seus atributos e infraestrutura. Você cria um modelo que descreve todos os recursos da

AWS que deseja (como chaves de acesso, grupos, políticas de grupo, perfis de instância, políticas gerenciadas, provedores de OIDC, políticas em linha, perfis, políticas de perfil, provedores de SAML, certificados de servidor, perfis vinculados a serviço, usuários (e adição de usuários a grupos), políticas de usuário e dispositivos virtuais de MFA), e o CloudFormation provisiona e configura esses recursos para você.

Quando você usa o CloudFormation, pode reutilizar o modelo para configurar os recursos do IAM repetidas vezes de modo consistente. Descreva os recursos uma vez e provisione os mesmos recursos repetidamente em várias regiões e Contas da AWS.

IAM e os modelos do CloudFormation

Para provisionar e configurar recursos para o IAM e os serviços relacionados, você deve entender os modelos do CloudFormation. Os modelos são arquivos de texto formatados em JSON ou YAML. Esses modelos descrevem os atributos que você deseja provisionar nas suas pilhas CloudFormation. Se não estiver familiarizado com o JSON ou o YAML, é possível usar o CloudFormation Designer para ajudar a começar a usar os modelos do CloudFormation. Para mais informações, consulte O que é o CloudFormation Designer? no Manual do usuário da AWS CloudFormation.

O IAM permite a criação de chaves de acesso, grupos, políticas de grupo, perfis de instância, políticas gerenciadas, provedores de OIDC, políticas em linha, perfis, políticas de perfil, provedores de SAML, certificados de servidor, perfis vinculados a serviço, usuários (e adição de usuários a grupos), políticas de usuário e dispositivos virtuais de MFA no CloudFormation. Para obter mais informações, incluindo exemplos de modelos JSON e YAML para os recursos do IAM, consulte Referência de tipos de recurso do AWS Identity and Access Management no Guia do usuário do AWS CloudFormation.

Você também pode criar modelos que criam os recursos relacionados, como perfis e políticas gerenciadas.

Saiba mais sobre o CloudFormation

Para saber mais sobre o CloudFormation, consulte os seguintes recursos:

- AWS CloudFormation
- AWS CloudFormation Guia do usuário do
- CloudFormation Referência de API do
- AWS CloudFormation Guia do Usuário da Interface de Linha de Comando

Usar o AWS CloudShell para operação com o AWS Identity and Access Management

AWS CloudShell O é um shell pré-autenticado baseado em navegador que você pode iniciar diretamente do Console de gerenciamento da AWS. É possível executar comandos da AWS CLI para serviços da AWS (incluindo o AWS Identity and Access Management) usando o shell de sua preferência (Bash, PowerShell ou Z shell). E você pode fazer isso sem precisar baixar ou instalar ferramentas de linha de comando.

Você inicia a AWS CloudShell via Console de gerenciamento da AWS, e as credenciais da AWS que usou para fazer login no console estarão automaticamente disponíveis em uma nova sessão do shell. Essa pré-autenticação de usuários da AWS CloudShell permite que você pule a configuração de credenciais ao interagir com serviços da AWS como o IAM usando a AWS CLI versão 2 (pré-instalada no ambiente computacional do shell).

Obter permissões do IAM para o AWS CloudShell

Usando os recursos de gerenciamento de acesso fornecidos pelo AWS Identity and Access Management, os administradores podem conceder permissões aos usuários do IAM para que eles possam acessar a AWS CloudShell e usar os recursos do ambiente.

A maneira mais rápida de um administrador conceder acesso aos usuários é por meio de uma política gerenciada pela AWS. Uma política gerenciada pela AWS é uma política independente que é criada e administrada pela AWS. A política gerenciada pela AWS a seguir para o CloudShell pode ser anexada às identidades do IAM:

- `AWSCloudShellFullAccess`: concede permissão para uso da AWS CloudShell com acesso total a todos os recursos.

Se você quiser limitar o escopo das ações que um usuário do IAM pode realizar com a AWS CloudShell, crie uma política personalizada que use a política gerenciada `AWSCloudShellFullAccess` como modelo. Para obter mais informações sobre como limitar as ações que estão disponíveis para os usuários no CloudShell, consulte Gerenciamento de acesso e uso da AWS CloudShell com políticas do IAM no Guia do usuário da AWS CloudShell.

Interagir com o IAM

Depois de iniciar a AWS CloudShell a partir do Console de gerenciamento da AWS, será possível começar imediatamente a interagir com o IAM usando a interface de linha de comando.

> **Note**
>
> Ao usar a AWS CLI na AWS CloudShell, não é necessário baixar nem instalar nenhum recurso adicional. Além disso, como você já está autenticado no shell, não precisará configurar as credenciais antes de fazer chamadas.

Criar um grupo do IAM e adicionar um usuário do IAM ao grupo usando um SDK AWS CloudShell

O exemplo a seguir usa o CloudShell para criar um grupo do IAM, adicionar um usuário do IAM ao grupo e verificar se o comando teve êxito.

1. A partir do Console de gerenciamento da AWS, é possível iniciar o CloudShell escolhendo opções a seguir disponíveis na barra de navegação:
   - Escolha o ícone do CloudShell.
   - Comece digitando “cloudshell” na caixa Pesquisar e escolha a opção CloudShell.
2. Para criar um grupo do IAM, insira o comando a seguir na linha de comando do CloudShell. Neste exemplo, chamamos o grupo de east_coast:

   ```
   aws iam create-group --group-name east_coast
   ```

   Se a chamada tiver êxito, a linha de comando exibirá uma resposta do serviço semelhante à seguinte saída:

   ```
   {
      "Group": {
          "Path": "/",
          "GroupName": "east_coast",
          "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
          "Arn": "arn:aws:iam::111122223333:group/east_coast",
          "CreateDate": "2023-09-11T21:02:21+00:00"
       }
   ```

```
        }
```

3. Para adicionar um usuário ao grupo que você criou, use o comando a seguir, especificando o nome do grupo e o nome de usuário. Neste exemplo, chamamos o grupo de east_coast e o usuário de john:

```
aws iam add-user-to-group --group-name east_coast --user-name john
```

4. Para verificar se o usuário está no grupo, use o comando a seguir, especificando o nome do grupo. Neste exemplo, continuamos a usar o grupo east_coast:

```
aws iam get-group --group-name east_coast
```

   Se a chamada tiver êxito, a linha de comando exibirá uma resposta do serviço semelhante à seguinte saída:

```
{
  "Users": [
    {
        "Path": "/",
        "UserName": "john",
        "UserId": "AIDAYBDBW4JBXGEXAMPLE",
        "Arn": "arn:aws:iam::552108220995:user/john",
        "CreateDate": "2023-09-11T20:43:14+00:00",
        "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
    }
  ],
  "Group": {
        "Path": "/",
        "GroupName": "east_coast",
        "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
        "Arn": "arn:aws:iam::111122223333:group/east_coast",
        "CreateDate": "2023-09-11T21:02:21+00:00"
     }
}
```

Using this service with an AWS SDK

AWSOs kits de desenvolvimento de software (SDKs) da estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que permitem que os desenvolvedores criem facilmente aplicações em seu idioma de preferência.

Documentação do SDK	Exemplos de código
AWS SDK para C++	AWS SDK para C++ Exemplos de código
AWS CLI	AWS CLI Exemplos de código da
AWS SDK para Go	AWS SDK para Go Exemplos de código da
AWS SDK para Java	AWS SDK para Java Exemplos de código da
AWS SDK para JavaScript	AWS SDK para JavaScript Exemplos de código da
AWS SDK para Kotlin	AWS SDK para Kotlin Exemplos de código da
AWS SDK para .NET	AWS SDK para .NET Exemplos de código da
AWS SDK para PHP	AWS SDK para PHP Exemplos de código da
Ferramentas da AWS para PowerShell	Ferramentas da AWS para PowerShell Exemplos de código da
AWS SDK para Python v4	Exemplos de códigos do AWS SDK para Python v4
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) Exemplos de código
AWS SDK para Ruby	AWS SDK para Ruby Exemplos de código da
AWS SDK para Rust	AWS SDK para Rust Exemplos de código da
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP Exemplos de código da

Documentação do SDK	Exemplos de código
AWS SDK para Swift	AWS SDK para Swift Exemplos de código do

Para obter exemplos específicos deste serviço, consulte Exemplos de código para o IAM usando AWS SDKs.

> **Exemplo de disponibilidade**
>
> Não consegue encontrar o que precisa? Solicite um exemplo de código usando o link Fornecer feedback na parte inferior desta página.

Informações de referência para o AWS Identity and Access Management

Use os tópicos nesta seção para localizar material de referência detalhado para diversos aspectos do IAM e do AWS STS.

Tópicos

- Identifique recursos da AWS com nomes do recurso da Amazon (ARNs)
- Identificadores do IAM
- IAM e cotas do AWS STS
- Suporte a endpoint de pilha dupla
- Endpoints da VPC de interface
- AWSServiços da que funcionam com o IAM
- AWS Signature Version 4 para solicitações de API
- Referência de política JSON do IAM

- Informações simplificadas sobre o AWS service (Serviço da AWS) para acesso programático

Identifique recursos da AWS com nomes do recurso da Amazon (ARNs)

Nomes de recurso da Amazon (ARNs) identificam apenas recursos da AWS. Nós exigimos um ARN quando você precisa especificar um recurso sem ambiguidade em toda a AWS, como em políticas do IAM, Amazon Relational Database Service (Amazon RDS), etiquetas e chamadas de API. Embora os ARNs, assim como todas as informações de identificação, devam ser usados e compartilhados com cuidado, eles não são considerados informações secretas, sigilosas ou confidenciais.

Formato ARN

A seguir estão os formatos gerais de ARNs. Os formatos específicos dependem do recurso. Para usar um ARN, substitua o texto em *itálico* pelas informações específicas do recurso. Lembre-se de que os ARNs para alguns recursos omitem a região, o ID da conta, ou a região e o ID da conta.

```
arn:partition:service:region:account-id:resource-id
```

```
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

A partição na qual o recurso está localizado. Uma partição é um grupo de regiões da AWS. Cada conta da AWS tem escopo para uma partição.

Estas são as partições compatíveis:

- `aws`: regiões da AWS
- `aws-cn`: regiões da China
- `aws-us-gov`: regiões da AWS GovCloud (US)

service

O namespace de serviço que identifica o produto da AWS.

region

O código da região. Por exemplo, `us-east-2` para Leste dos EUA (Ohio). Para obter uma lista de códigos de região, consulte Endpoints regionais na Referência geral da AWS.

account-id

O ID da conta da AWS que possui o recurso, sem hifens. Por exemplo, `123456789012`.

resource-type

O tipo de recurso. Por exemplo, a `vpc` para uma nuvem privada virtual (VPC).

resource-id

O identificador do recurso. É o nome do recurso, a ID do recurso ou o caminho do recurso. Alguns identificadores de recursos incluem um recurso pai (sub-resource-type/parent-resource/sub-resource) ou um qualificador, como uma versão (resource-type:resource-name:qualifier).

Exemplos

IAM user (Usuário do IAM)

arn:aws:iam::*123456789012*:user/*john*

Tópico do SNS

arn:aws:sns:*us-east-1*:*123456789012*:*example-sns-topic-name*

VPC

arn:aws:ec2:*us-east-1*:*123456789012*:vpc/*vpc-0e9801d129EXAMPLE*

Encontrar o formato do ARN de um recurso

O formato exato de um ARN depende do serviço e do tipo de recurso. Alguns ARNs de recursos podem incluir um caminho, uma variável ou um caractere curinga. Para encontrar o formato do ARN de um recurso da AWS específico, abra a Referência de autorização de serviços, abra a página do serviço e navegue até a tabela de tipos de recurso.

Caminhos em ARNs

Os ARNs de recursos podem incluir um caminho. Por exemplo, no Amazon S3, o identificador do recurso é um nome de objeto que pode incluir barras (/) para formar um caminho. Da mesma forma, nomes de usuários e nomes de grupo do IAM podem incluir caminhos. Somente caracteres alfanuméricos e os seguintes caracteres gráficos são permitidos nos caminhos do IAM: barra (/), mais (+), igual (=), vírgula (,), ponto final (.), arroba (@), sublinhado (_) e hífen (-).

Usar curingas em caminhos

Os caminhos podem incluir um caractere curinga, ou seja, um asterisco (*). Alguns elementos da política permitem curingas, enquanto outros não. Você pode usar curingas para os elementos Resource ou NotResource, mas não para os elementos Principal ou NotPrincipal. Para obter mais informações, consulte Referência de política JSON do IAM.

É possível especificar `role/*` para significar todas as funções na conta 123456789012, como no seguinte exemplo:

```
arn:aws:iam::123456789012:role/*
```

Também é possível pode finalizar o nome de um recurso com um caractere curinga. Por exemplo, é possível especificar **`service-*`** para significar todos os perfis que começam com `service` e terminam com caracteres diferentes, como `service-role1` ou `service-test`:

```
arn:aws:iam::123456789012:role/service-*
```

O exemplo a seguir mostra ARNs para objetos em um bucket do Amazon S3 em que o nome do recurso inclui um caminho. O ARN `arn:aws:s3:::amzn-s3-demo-bucket/*` aplica-se a todos

os objetos dentro desse bucket, independentemente do prefixo. O ARN `arn:aws:s3:::amzn-s3-demo-bucket/Development/*` aplica-se a todos os objetos criados dentro do prefixo **`/Development/`**.

Você também pode usar o caractere curinga ? para especificar um caractere em um ARN. Por exemplo, é possível usar o ARN a seguir para todas as pastas que começam com quatro caracteres e terminam em **`-test`** no bucket do S3 chamado amzn-s3-demo-bucket. Algumas pastas que corresponderiam a isso incluem `1234-test`, `2024-test` ou `a100-test`.

```
arn:aws:s3:::amzn-s3-demo-bucket/????-test
```

Também é possível usar curingas nas diferentes seções de um ARN, delimitados por dois pontos ":". No exemplo a seguir, dois curingas são usados para combinar todos os aplicativos e recursos do Amazon Q dentro dos aplicativos em todas as regiões para a conta 123456789012:

```
arn:aws:qbusiness:*:123456789012:*
```

Da mesma forma, o exemplo a seguir corresponde a todas as Amazon VPCs em todas as regiões para a conta 123456789012:

```
arn:aws:ec2:*:123456789012:vpc/*
```

Da mesma forma, o exemplo a seguir corresponde a todos os volumes do Amazon EBS em todas as regiões para a conta 123456789012:

```
arn:aws:ec2:*:123456789012:volume/*
```

Limitações no uso de curingas em ARNs

Não é possível usar curingas na parte do ARN que especifica o tipo de recurso. O seguinte exemplo de ARN com um caractere curinga dentro do tipo de recurso não é válido:

```
arn:aws:lambda:us-east-2:123456789012:functi*:my-function <== not allowed
```

> **Note**
>
> Quando você especifica um ARN incompleto (um com menos do que os seis campos padrão) em uma política baseada em identidade, a AWS preenche automaticamente

o ARN adicionando caracteres curinga (*) a todos os campos ausentes. Por exemplo, especificar `arn:aws:sqs` é equivalente a `arn:aws:sqs:*:*:*`, o que concede acesso a todos os recursos do Amazon SQS em todas as regiões e contas. No entanto, as políticas de sessão passadas para as solicitações do AWS STS AssumeRole, AssumeRoleWithWebIdentity e AssumeRoleWithSAML não oferecem suporte a ARNs incompletos. Usar um ARN incompleto em uma política de sessão resultará em um erro `MalformedPolicyDocumentException`.

Também não é possível usar curingas no ARN do prefixo nem ter curingas na seção de partição de um ARN.

```
arn:aws:redshift:us-east-1:123456789012:? <== not allowed
```

Identificadores do IAM

O IAM usa alguns identificadores diferentes para usuários, perfis, políticas, certificados de servidor e grupos do IAM. Esta seção descreve os identificadores e quando usar cada um deles.

Tópicos

- Nomes amigáveis e caminhos
- ARNs do IAM
- Identificadores exclusivos
- Noções básicas sobre prefixos exclusivos de IDs
- Obter o identificador exclusivo

Nomes amigáveis e caminhos

Ao criar um usuário, uma função, um grupo de usuários ou uma política, ou ao carregar um certificado de servidor, você atribui a ele um nome amigável. Exemplos incluem Bob, TestApp1, Desenvolvedores, ManageCredentialsPermissions ou ProdServerCert.

Se você usar a API do IAM ou a AWS Command Line Interface (AWS CLI) para criar recursos do IAM, poderá adicionar um caminho opcional. Você pode usar um único caminho ou aninhar vários caminhos como se fossem uma estrutura de pastas. Por exemplo, você pode usar o caminho aninhado `/division_abc/subdivision_xyz/product_1234/engineering/` de acordo com

a estrutura organizacional da empresa. Em seguida, você pode criar uma política para permitir que todos os usuários no caminho acessem a API do simulador de políticas. Para exibir esta política, consulte: IAM: acessar a API do simulador de política com base no caminho do usuário. Para obter informações sobre como especificar um nome simples, consulte a documentação da API do usuário. Para obter mais exemplos de como você pode usar caminhos, consulte ARNs do IAM.

Quando você usa o CloudFormation para criar recursos, pode especificar um caminho para os usuários, os perfis e os grupos do IAM, e para as políticas gerenciadas pelo cliente.

Se você tiver um usuário e um grupo de usuários no mesmo caminho, o IAM não colocará o usuário automaticamente nesse grupo de usuários. Por exemplo, você pode criar um grupo de usuários Desenvolvedores e especificar o caminho como `/division_abc/subdivision_xyz/product_1234/engineering/`. Se você criar um usuário chamado Bob e adicionar o mesmo caminho a ele, isso não colocará Bob automaticamente no grupo de usuários desenvolvedores. O IAM não impõe nenhuma barreira entre usuários ou grupos do IAM com base em seus caminhos. Usuários com caminhos diferentes podem usar os mesmos recursos se tiverem permissão para esses recursos. O número e o tamanho dos recursos do IAM em uma conta da AWS são limitados. Para obter mais informações, consulte IAM e cotas do AWS STS.

ARNs do IAM

A maioria dos recursos tem um nome amigável, por exemplo, um usuário chamado `Bob` ou um grupo de usuários chamado `Developers`. No entanto, a linguagem da política de permissões exige que você especifique o recurso ou recursos usando o seguinte formato de nome de recurso da Amazon (ARN).

```
arn:partition:service:region:account:resource
```

Em que:

- `partition` identifica a partição do recurso. Para regiões padrão da AWS a partição é `aws`. Se você tem recursos em outras partições, a partição é `aws-`*`partitionname`*. Por exemplo, a partição de recursos na região China (Pequim) é `aws-cn`. Não é possível delegar acesso entre contas em partições diferentes.
- `service` identifica o produto da AWS. Os recursos do IAM sempre usam `iam`.
- `region` identifica a região do recurso. Para recursos do IAM, ela é sempre deixada em branco.
- `account` especifica o ID da Conta da AWS sem hifens.
- `resource` identifica o recurso específico pelo nome.

Você pode especificar ARNs do IAM e do AWS STS usando a sintaxe a seguir. A parte da região do ARN está em branco porque os recursos do IAM são globais.

Sintaxe:

```
arn:aws:iam::account:root
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
arn:aws:iam::aws:contextProvider/context-provider-name
```

Muitos dos seguintes exemplos incluem caminhos na parte de recurso do ARN. Caminhos não podem ser criados ou manipulados no Console de gerenciamento da AWS. Para usar caminhos, você deve trabalhar com o recurso usando a API da AWS, a AWS CLI ou o Tools for Windows PowerShell.

Exemplos:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/John
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/John
```

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Jane
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneMFA
arn:aws:iam::123456789012:u2f/user/John/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
arn:aws:iam::aws:contextProvider/IdentityCenter
```

Os exemplos a seguir fornecem mais detalhes para ajudar você a entender o formato do ARN para diferentes tipos de recursos do IAM e do AWS STS.

- Um usuário do IAM na conta:

> **Note**
>
> O nome de cada usuário do IAM é exclusivo. O nome de usuário não diferencia maiúsculas de minúsculas em situações como durante o processo de login, mas diferencia maiúsculas de minúsculas quando você o usa em uma política ou como parte de um ARN.

```
arn:aws:iam::123456789012:user/John
```

- Outro usuário com um caminho que reflete um gráfico da organização:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/Jane
```

- Um grupo de usuários do IAM:

```
arn:aws:iam::123456789012:group/Developers
```

- Um grupo de usuários do IAM com um caminho:

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Uma função do IAM:

```
arn:aws:iam::123456789012:role/S3Access
```

- Uma função vinculada a serviços:

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Uma função de serviço:

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Uma política gerenciada:

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Um perfil de instância que pode ser associado à uma instância do Amazon EC2:

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Uma entidade principal de usuário federado do AWS STS identificada no IAM como "Paulo":

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- A sessão ativa de alguém assumindo a função de "Accounting-Role", com um nome de sessão de função "Mary":

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- Representa a própria sessão do chamador quando usada como recurso em uma chamada de API, como a API AWS STS SetContext, que opera na sessão de chamada:

```
arn:aws:sts::123456789012:self
```

- O dispositivo de autenticação multifator atribuído ao usuário chamado Jorge:

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Um certificado de servidor:

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Um certificado de servidor com um caminho que reflete um gráfico da organização:

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Provedores de identidade (SAML e OIDC):

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- Provedor de identidade OIDC com um caminho que reflete um URL do provedor de identidade OIDC do Amazon EKS:

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

- O ARN do provedor de contexto usado com a propagação de identidade confiável do Centro de Identidade do AWS IAM a partir do qual a declaração de contexto confiável foi gerada:

```
arn:aws:iam::aws:contextProvider/IdentityCenter
```

Outro ARN importante é o ARN do usuário raiz. Embora esse não seja um recurso do IAM, você deve estar familiarizado com o formato desse ARN. Ele costuma ser usado no elemento `Principal` de uma política baseada em recursos.

- A Conta da AWS exibe o seguinte:

```
arn:aws:iam::123456789012:root
```

O exemplo a seguir mostra uma política que você pode atribuir a Richard para permitir que ele gerencie suas próprias chaves de acesso. Observe que o recurso é o usuário do IAM Richard.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/
Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Ao usar ARNs para identificar recursos em uma política do IAM, você pode incluir variáveis de política. As variáveis de política podem incluir espaços reservados para informações de runtime (como o nome do usuário) como parte do ARN. Para obter mais informações, consulte Elementos de política do IAM: variáveis e etiquetas

Uso de curingas e caminhos em ARNs

Você pode usar caracteres curinga na parte *recurso* do ARN para especificar vários usuários, políticas ou grupos do IAM. Por exemplo, para especificar todos os usuários trabalhando em product_1234, você usa:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Se tiver usuários cujos nomes começam com a string app_, você pode se referir a todos eles com o seguinte ARN:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Para especificar todos os usuários, políticas ou grupos do IAM da sua Conta da AWS, use um caractere curinga após a parte `user/`, `group/` ou `policy/` do ARN, respectivamente.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Se você especificar o seguinte ARN para um usuário `arn:aws:iam::111122223333:user/*`, ele corresponderá a ambos os exemplos a seguir.

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Mas, se você especificar o ARN a seguir para um usuário `arn:aws:iam::111122223333:user/division_abc*`, ele corresponderá ao segundo exemplo, mas não ao primeiro.

```
arn:aws:iam::111122223333:user/John
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Jane
```

Não use um curinga na parte `user/`, `group/` ou `policy/` do ARN. Por exemplo, o IAM não permite o seguinte:

```
arn:aws:iam::123456789012:u*
```

> **Note**
>
> Quando você especifica um ARN incompleto (um com menos do que os seis campos padrão) em uma política baseada em identidade, a AWS preenche automaticamente o ARN adicionando caracteres curinga (*) a todos os campos ausentes. Por exemplo, especificar `arn:aws:sqs` é equivalente a `arn:aws:sqs:*:*:*`, o que concede acesso a todos os recursos do Amazon SQS em todas as regiões e contas. No entanto, as políticas de sessão passadas para as solicitações do AWS STS AssumeRole, AssumeRoleWithWebIdentity e AssumeRoleWithSAML não oferecem suporte a ARNs incompletos. Usar um ARN incompleto em uma política de sessão resultará em um erro `MalformedPolicyDocumentException`.

Example Exemplo de uso de caminhos e ARNs para um grupo de usuários baseado em projeto

Caminhos não podem ser criados ou manipulados no Console de gerenciamento da AWS. Para usar caminhos, você deve trabalhar com o recurso usando a API da AWS, a AWS CLI ou o Tools for Windows PowerShell.

Neste exemplo, Jules do grupo de usuários Marketing_Admin cria um grupo de usuários baseado em projeto no caminho /marketing/. Jules atribui usuários de diferentes partes da empresa ao grupo de usuários. Este exemplo mostra que o caminho de um usuário não está relacionado aos grupos de usuários em que o usuário está.

O grupo de marketing tem um novo produto para lançamento, portanto, Jules cria um novo grupo de usuários no caminho /marketing/ chamado Widget_Launch. Jules então atribui a seguinte política ao grupo de usuários, que fornece ao grupo de usuários acesso a objetos na parte do `example_bucket` designada para esse lançamento em particular.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/
*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/
widget/*"}}
    }
  ]
}
```

Jules então atribui os usuários que estão trabalhando neste lançamento ao grupo de usuários. Isso inclui Patrícia e Eli do caminho /marketing/. E também inclui Chris e Chloe do caminho /sales/ e Alice e Jim do caminho /legal/.

Identificadores exclusivos

Quando o IAM cria um usuário, grupo de usuários, função, política, perfil de instância ou certificado de servidor, ele atribui a cada recurso um ID exclusivo. O ID exclusivo é semelhante a:

```
AIDAJQABLZS4A3QDU576Q
```

Na maioria dos casos, use nomes amigáveis e ARNs ao trabalhar com recursos do IAM. Dessa forma, você não precisa saber o ID exclusivo de um recurso específico. No entanto, o ID exclusivo às vezes pode ser útil quando não é prático usar nomes amigáveis.

Um exemplo reutiliza nomes amigáveis na sua Conta da AWS. Dentro de sua conta, um nome amigável para um usuário, um grupo de usuários, função ou uma política deve ser exclusivo. Por exemplo, você pode criar um usuário do IAM chamado `John`. Sua empresa usa o Amazon S3 e tem um bucket com pastas para cada funcionário. O usuário do IAM `John` é membro de um grupo de usuários do IAM chamado `User-S3-Access` com permissões que permitem aos usuários acessar apenas suas próprias pastas no bucket. Para obter um exemplo de como você pode criar uma política baseada em identidade que permita que usuários do IAM acessem seu próprio objeto de bucket no S3 usando o nome amigável de usuários, consulte Amazon S3: permite que usuários do IAM acessem seus diretórios base do S3 de forma programática e no console.

Suponha que o funcionário chamado John deixe sua empresa e você exclua o usuário do IAM correspondente chamado `John`. Posteriormente, outro funcionário chamado John começa a trabalhar e você cria um novo usuário do IAM chamado `John`. Você adiciona o novo usuário do IAM chamado `John` ao grupo de usuários do IAM `User-S3-Access` existente. Se a política associada ao grupo de usuários especificar o nome amigável de usuário do IAM `John`, ela permitirá que o novo John acesse informações deixadas pelo antigo John.

Em geral, recomendamos que você especifique o ARN do recurso em suas políticas em vez de seu ID exclusivo. No entanto, todo usuário do IAM tem um ID exclusivo, mesmo se você criar um novo usuário do IAM que reutilize um nome amigável que você excluiu antes. No exemplo, o antigo usuário do IAM `John` e o novo usuário do IAM `John` têm IDs exclusivos diferentes. Você pode criar políticas baseadas em recurso que concedem acesso por ID exclusivo e não apenas por nome de usuário. Isso reduz a chance de você, inadvertidamente, conceder acesso a informações que um funcionário não deve ter.

O exemplo a seguir mostra como é possível especificar IDs exclusivos no elemento `Principal` de uma política baseada em recursos.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

O exemplo a seguir mostra como é possível especificar IDs exclusivos no elemento `Condition` de uma política usando uma chave de condição global `aws:userid`.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Outro exemplo em que IDs de usuário podem ser úteis é se você mantiver seu próprio banco de dados (ou outro armazenamento) de informações de perfil ou de usuário do IAM. O ID exclusivo pode fornecer um identificador exclusivo para cada perfil ou usuário do IAM criado. Esse é o caso quando há perfis ou usuários do IAM que reutilizam um nome, como no exemplo anterior.

Noções básicas sobre prefixos exclusivos de IDs

O IAM usa os prefixos a seguir para indicar a qual tipo de recurso cada ID exclusivo se aplica. Os prefixos podem variar com base em quando foram criados.

Prefixo	Tipo de recurso
ABIA	AWS STS Token de portador de serviço do
ACCA	Credencial específica de contexto
AGPA	Grupo de usuários
AIDA	IAM user (Usuário do IAM)

Prefixo	Tipo de recurso
AIPA	Perfil de instância do Amazon EC2
AKIA	Chave de acesso
ANPA	Política gerenciada
ANVA	Versão em uma política gerenciada
APKA	Chave pública
AROA	Perfil
ASCA	Certificado
ASIA	IDs de chave de acesso temporárias (AWS STS) usam esse prefixo, mas são exclusivas somente em combinação com a chave de acesso secreta e o token da sessão.

Obter o identificador exclusivo

O ID exclusivo para um recurso do IAM não está disponível no console do IAM. Para obter o ID exclusivo, você pode usar os seguintes comandos da AWS CLI ou chamadas de API do IAM.

AWS CLI:

- get-caller-identity
- get-group
- get-role
- get-user
- get-policy
- get-instance-profile
- get-server-certificate

API do IAM:

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM e cotas do AWS STS

O AWS Identity and Access Management (IAM) e o AWS Security Token Service (STS) têm cotas que limitam o tamanho dos objetos. Isso afeta como um objeto é nomeado, o número de objetos que podem ser criados e o número de caracteres que podem ser usados ao transmitir um objeto.

> **Note**
>
> Para obter informações no nível da conta sobre o uso e as cotas do IAM use a operação da API GetAccountSummary ou o comando da AWS CLI get-account-summary.

Requisitos de nome do IAM

Os nomes do IAM têm os seguintes requisitos e restrições:

- Os documentos de política podem conter apenas os seguintes caracteres Unicode: guia horizontal (U+0009), linefeed (U+000A), retorno de carro (U+000D) e caracteres no intervalo de U+0020 a U +00FF.
- Os nomes de usuários, grupos, funções, políticas, perfis da instância e certificados de servidor devem ser alfanuméricos, incluindo os seguintes caracteres comuns: mais (+), igual (=), vírgula (,), ponto final (.), arroba (@), sublinhado (_) e hífen (-). Os nomes de caminhos devem começar e terminar com uma barra (/).
- Os nomes de usuários, grupos, funções e perfis de instância devem ser exclusivos na conta. Eles não são diferenciados por maiúsculas e minúsculas, por exemplo, você não pode criar dois grupos denominados **`ADMINS`** e **`admins`**.

- O valor do ID externo que terceiros usam para assumir uma função deve ter no mínimo 2 e no máximo 1.224 caracteres. O valor deve ser alfanumérico sem espaço em branco. Ele também pode incluir os seguintes símbolos: mais (+), igual (=), vírgula (,), ponto (.), arroba (@), dois pontos (:), barra (/) e hífen (-). Para obter mais informações sobre o ID externo, consulte Acesso às Contas da AWS de propriedade de terceiros.
- Os nomes de políticas para as políticas em linha devem ser exclusivos para o usuário, grupo ou perfil em que eles são incorporados. Os nomes podem conter caracteres latinos básicos (ASCII), exceto os seguintes caracteres reservados: barra invertida (\), barra (/), asterisco (*), ponto de interrogação (?) e espaço em branco. Esses caracteres são reservados de acordo com a RFC 3986, seção 2.2.
- As senhas de usuário (perfis de login) podem conter caracteres latinos básicos (ASCII).
- Os alias do ID da Conta da AWS devem ser exclusivos entre os produtos da AWS e devem ser alfanuméricos, seguindo as convenções de nomeação do DNS. Um alias deve ser em letras minúsculas, não deve iniciar ou terminar com um hífen, não pode conter dois hifens consecutivos e não pode ser um número de 12 dígitos.

Para obter uma lista de caracteres latinos básicos (ASCII), consulte a Tabela de códigos latinos básicos (ASCII) da Biblioteca do Congresso.

Cotas de objetos do IAM

As cotas, também conhecidas como limites na AWS, são os valores máximos para recursos, ações e itens na sua Conta da AWS. Use o Service Quotas para gerenciar as cotas do IAM.

Para ver a lista de endpoints e cotas de serviço do IAM, consulte Endpoints e cotas do AWS Identity and Access Management no Referência geral da AWS.

Para solicitar um aumento da cota

1. Siga o procedimento de login adequado para o tipo de usuário, conforme descrito no tópico Como iniciar sessão na AWS no Guia do usuário do AWS Sign-In para entrar no Console de gerenciamento da AWS.
2. Abra o console do Service Quotas.
3. No painel de navegação, escolha Serviços da AWS.
4. Na barra de navegação, selecione a região US East (N. Virginia). Depois, procure **`IAM`**.
5. Selecione AWS Identity and Access Management (IAM), escolha uma cota e siga as instruções para solicitar um aumento de cota.

Para obter mais informações, consulte Solicitar um aumento de cota no Guia do usuário do Service Quotas.

Para ver um exemplo de como solicitar um aumento de cotas do IAM usando o console do Service Quotas, assista ao vídeo a seguir.

Solicitar um aumento de cotas do IAM usando o console do Service Quotas.

Você pode solicitar um aumento nas cotas padrão para cotas ajustáveis do IAM. Solicitações até o maximum quota são automaticamente aprovadas e são concluídas em poucos minutos.

A tabela a seguir lista os recursos para os quais os aumentos de cota podem ser aprovados automaticamente.

Recurso	Cota padrão	Cota máxima
Políticas gerenciadas pelo cliente por conta	1500	10000
Grupos por conta	300	500
Perfis de instância por conta	1000	10000
Políticas gerenciadas por perfil	20	25
Políticas gerenciadas por usuário	10	20
Políticas gerenciadas por grupo	10	10
Duração da política de confiança da função	2048 caracteres	8192 caracteres
Funções por conta	1000	10000
Certificados de servidor por conta	20	20
Provedores de conexão do OpenId por conta	100	700

Cotas do IAM Access Analyzer

Para ver a lista de endpoints e cotas de serviço do IAM Access Analyzer, consulte Endpoints e cotas do IAM Access Analyzer no Referência geral da AWS.

Cotas do IAM Roles Anywhere

Para ver a lista de endpoints e cotas de serviço do IAM Roles Anywhere, consulte Endpoints e cotas do serviço AWS Identity and Access Management Roles Anywhere no Referência geral da AWS.

Cotas de solicitações do STS

O AWS Security Token Service (AWS STS) impõe as cotas de solicitações a seguir.

Para solicitações ao AWS STS feitas com credenciais da AWS, a cota de solicitações padrão é de 600 solicitações por segundo, por conta, por região. As seguintes operações do AWS STS compartilham essa cota:

- AssumeRole
- DecodeAuthorizationMessage
- GetAccessKeyInfo
- GetCallerIdentity
- GetFederationToken
- GetSessionToken

> **Note**
>
> As solicitações feitas ao AWS STS por entidades principais de serviço da AWS, como as solicitações usadas para assumir perfis para uso com um serviço da AWS, não consomem a cota do STS de solicitações por segundo das suas contas.

Por exemplo, se uma Conta da AWS fizer 100 solicitações GetCallerIdentity por segundo e 100 chamadas AssumeRole por segundo na mesma região, essa conta estará consumindo 200 das 600 solicitações do STS disponíveis por segundo para essa região.

Para solicitações AssumeRole entre contas, somente a conta que faz a solicitação AssumeRole afeta a cota do STS. A cota da conta de destino não é consumida.

Para solicitar um aumento das cotas de solicitações do STS, abra um tíquete com o suporte da AWS.

> **Note**
>
> Com as próximas alterações no endpoint global do AWS STS (`https://sts.amazonaws.com`), as solicitações ao endpoint global não compartilharão uma cota de solicitações por segundo (RPS) com endpoints regionais do AWS STS em regiões habilitadas por padrão. Quando uma solicitação para o endpoint global do AWS STS é proveniente de uma única região, ela é contabilizada na cota de RPS do endpoint global. No entanto, quando as solicitações são provenientes de várias regiões, cada região adicional receberá sua própria cota de solicitações por segundo independente. Para obter mais informações sobre alterações em endpoints globais do AWS STS, consulte Alterações nos endpoints globais do AWS STS.

Limites de caracteres do IAM e do STS

Veja a seguir as contagens máximas de caracteres e limites de tamanho para IAM e o AWS STS. Você não pode solicitar um aumento dos limites a seguir.

Descrição	Limite
Alias de um ID de Conta da AWS	3 a 63 caracteres
Para políticas em linha	Você pode adicionar quantas políticas em linha desejar a um usuário, função ou grupo da IAM. No entanto, o tamanho total de política agregado (tamanho total de todas as políticas em linha) por entidade não pode exceder os seguintes limites: • O tamanho da política de usuário não pode exceder 2.048 caracteres. • O tamanho da política de perfil não pode exceder 10.240 caracteres. • O tamanho da política de grupo não pode exceder 5.120 caracteres.

Descrição	Limite
	Note O IAM; não conta espaços em branco ao calcular o tamanho de uma política em relação a esses limites.
Para políticas gerenciadas pelo cliente	• O tamanho de cada política gerenciada não pode exceder 6,144 caracteres. **Note** O IAM; não conta espaços em branco ao calcular o tamanho de uma política em relação a esse limite.
Nome de grupo	128 caracteres
Nome do perfil de instância	128 caracteres
Senha para um perfil de login	1 a 128 caracteres
Caminho	512 caracteres
Nome da política	128 caracteres
Nome da função	64 caracteres **Important** No entanto, se você pretende usar um perfil com o recurso Alternar perfil no Console de gerenciamento da AWS, `Path` e `RoleName` combinados não podem exceder 64 caracteres.

Descrição	Limite
Duração da sessão da função	12 horas Quando você assume uma função a partir da AWS CLI ou da API, pode usar o parâmetro da CLI `duration-seconds` ou o parâmetro da API `DurationSeconds` para solicitar uma sessão de função mais longa. É possível especificar um valor de 900 segundos (15 minutos) até a configuração da duração máxima da sessão para a função, que pode variar de 1 a 12 horas. Se você não especificar um valor para o parâmetro `DurationSeconds`, as credenciais de segurança serão válidas por uma hora. Os usuários do IAM que trocam de perfis no console recebem a duração máxima da sessão ou o tempo restante na sessão do usuário, o que for menor. A configuração da duração máxima da sessão não limita as sessões assumidas por serviços da AWS. Para saber como visualizar o valor máximo para sua função, consulte Atualizar a duração máxima da sessão de um perfil.
Nome da sessão da função	64 caracteres

Descrição	Limite
Políticas de sessão da função	• O tamanho do documento de política JSON passado e todos os caracteres ARN de política gerenciada passados combinados não podem exceder 2.048 caracteres. • Você pode passar, no máximo, 10 ARNs de política gerenciada ao criar uma sessão. • É possível passar apenas um documento de política JSON ao criar uma sessão temporária de forma programática para um perfil ou uma entidade principal de usuário federado do AWS STS. • Além disso, as políticas de sessão e as tags de sessão aumentam o tamanho do token da sessão, que tem um limite específico. O elemento de resposta `SessionTokenUtilization` mostra qual porcentagem do limite de tamanho o token usa. O elemento de resposta `PackedPolicySize` está obsoleto. Use `SessionTokenUtilization` em vez disso. • Recomendamos que você passe políticas de sessão usando a AWS CLI ou API da AWS. O Console de gerenciamento da AWS pode aumentar o tamanho do token da sessão adicionando informações da sessão.

Descrição	Limite
Tags de sessão da função	• As tags de sessão devem atender ao limite de chave de tag de 128 caracteres e ao limite de valor de tag de 256 caracteres. • Você pode passar até 50 tags de sessão. • As políticas de sessão e as tags de sessão aumentam o tamanho do token da sessão, que tem um limite específico. Você pode passar tags de sessão usando o AWS CLI ou API da AWS. O elemento de resposta `SessionTokenUtilization` mostra qual porcentagem do limite de tamanho o token usa. O elemento de resposta `PackedPolicySize` está obsoleto. Use `SessionTokenUtilization` em vez disso.
Resposta de autenticação SAML codificada em base64	100.000 caracteres Esse limite de caracteres se aplica à CLI `assume-role-with-saml` ou à operação de API `AssumeRoleWithSAML`.
Chave de tag	128 caracteres Esse limite de caracteres se aplica a tags em recursos do IAM e tags de sessão.
Valor da tag	256 caracteres Esse limite de caracteres se aplica a tags em recursos do IAM e tags de sessão. Os valores da etiqueta podem estar vazios, o que significa que os valores da etiqueta podem ter 0 caractere.

Descrição	Limite
IDs únicos criados pelo IAM	128 caracteres. Por exemplo: • IDs de usuários que começam com `AIDA` • IDs de grupos que começam com `AGPA` • IDs de função que começam com `AROA` • IDs de políticas gerenciadas que começam com `ANPA` • IDs de certificado de servidor que começam com `ASCA` **Note** Isso não é destinado a ser uma lista completa, nem é uma garantia de que os IDs de um determinado tipo começam apenas com a combinação de letras especificadas.
Nome de usuário	64 caracteres

Suporte a endpoint de pilha dupla

Um endpoint de pilha dupla permite que os clientes se comuniquem com os serviços da AWS usando endereços IPv4 ou IPv6. Tanto o IAM como o AWS STS fornecem suporte para o endpoint de pilha dupla. Para obter mais informações sobre como configurar a VPC para o modo de pilha dupla, consulte Suporte a IPv6 para sua VPC no Guia do usuário da Amazon VPC.

Se você usa políticas do IAM que incluem as chaves de condição `aws:sourceIp` ou `aws:vpcSourceIp`, você precisa atualizar essas políticas para oferecer suporte a clientes IPv6. Para obter mais informações sobre endereçamento IPv6 para suas VPCs, consulte Endereçamento IP para suas VPCs e sub-redes no Guia do usuário da Amazon VPC.

Suporte para endpoint com o modo dual stack no IAM

O IAM fornece um endpoint público de pilha dupla que é compatível com clientes IPv4 e IPv6. O endpoint público de pilha dupla do IAM também pode ser acessado de forma privada por meio de sua nuvem privada virtual (VPC) usando AWS PrivateLink. Para obter mais informações sobre a criação de endpoints da VPC de interface privada para o IAM, consulte Criar um VPC endpoint para o IAM.

O endpoint público de pilha dupla do IAM é `https://iam.global.api.aws`.

O endpoint público do IAM em `https://iam.amazonaws.com`, diferentemente do endpoint público de pilha dupla, é compatível somente com clientes IPv4. Quando acessado de forma privada de sua VPC usando AWS PrivateLink, o endpoint público do IAM é compatível com clientes IPv4 e IPv6.

Suporte ao endpoint de pilha dupla do AWS STS

O AWS STS fornece endpoints regionais de pilha dupla que são compatíveis com clientes IPv4 e IPv6. Os endpoints regionais de pilha dupla do AWS STS também podem ser acessados de forma privada por meio de sua nuvem privada virtual (VPC) usando o AWS PrivateLink. Para obter mais informações sobre a criação de endpoints da VPC para o AWS STS, consulte Criar um VPC endpoint para o AWS STS.

A tabela a seguir mostra os endpoints regionais de pilha dupla do AWS STS por partição:

Partition	URL do endpoint de pilha dupla
Regiões comerciais da AWS	Endpoints regulares: `https://sts.` *`<region>`*`.api.aws` Endpoints FIPS: `https://sts-fips.` *`<region>`*`.api.aws`
Regiões do AWS GovCloud (US)	Endpoints regulares: `https://sts.` *`<region>`*`.api.aws`
Regiões da China	Endpoints regulares: `https://sts.` *`<region>`*`.api.amazonwebservices.com.cn`

> **Note**
>
> Os endpoints FIPS se aplicam somente às regiões dos EUA e Canadá.

Limitações

As seguintes limitações se aplicam ao suporte do AWS STS a endpoints de pilha dupla:

- Os clientes IPv6 não são compatíveis com o endpoint global `https://sts.amazonaws.com`. Você deve usar endpoints regionais de pilha dupla para dar suporte a clientes IPv6.
- Os endpoints da VPC somente IPv6 do AWS STS não são compatíveis. Os endpoints da VPC podem ser configurados para conectividade IPv4 ou de pilha dupla.

Endpoints da VPC de interface

Se você usar o Amazon Virtual Private Cloud (Amazon VPC) para hospedar os recursos da AWS, poderá estabelecer uma conexão privada entre a VPC e o AWS Identity and Access Management (IAM) ou o AWS Security Token Service (AWS STS). Você pode usar essa conexão para permitir que o IAM ou o AWS STS se comunique com os seus recursos na VPC sem passar pela Internet pública.

A Amazon VPC é um produto da AWS que pode ser utilizado para iniciar os recursos da AWS em uma rede virtual definida por você. Com a VPC, você tem controle sobre as configurações de rede, como o intervalo de endereços IP, sub-redes, tabelas de rotas e gateways de rede. Para conectar sua VPC ao IAM ou ao AWS STS, defina um endpoint de interface da VPC para cada serviço. O endpoint fornece uma conectividade confiável e escalável ao IAM ou ao AWS STS sem a necessidade de um gateway da Internet, de uma instância de conversão de endereço de rede (NAT) ou de uma conexão VPN. Para obter mais informações, consulte O que é o Amazon VPC? no Guia do usuário do Amazon VPC.

Os endpoints da VPC de interface são habilitados pelo AWS PrivateLink, uma tecnologia da AWS viabiliza a comunicação privada entre os serviços da AWS usando uma interface de rede elástica com endereços IP privados. Para obter mais informações, consulte AWS PrivateLink para serviços da AWS.

As informações a seguir destinam-se aos usuários do Amazon VPC. Para obter mais informações, consulte Conceitos básicos do Amazon VPC no Guia do usuário do Amazon VPC.

Tópicos

- Disponibilidade de endpoint da VPC
- Criar um VPC endpoint para o IAM
- Criar um VPC endpoint para o AWS STS
- Criar um endpoint da VPC para descoberta do OIDC do AWS STS
- Controle de acesso ao AWS STS com as políticas de endpoint da VPC
- Uso de chaves de condição da VPC para controlar o acesso federado

Disponibilidade de endpoint da VPC

> **Important**
>
> Os endpoints da VPC para o IAM podem ser criados somente na região em que o ambiente de gerenciamento do IAM está localizado. Se a sua VPC estiver em uma região diferente da região do ambiente de gerenciamento do IAM, será necessário usar o AWS Transit Gateway para permitir acesso ao endpoint da VPC de interface do IAM de outra região. Para obter mais informações, consulte Criar um VPC endpoint para o IAM.

Atualmente, o IAM oferece suporte a endpoints da VPC nas seguintes regiões:

- Leste dos EUA (Norte da Virgínia)
- China (Pequim)
- AWS GovCloud (Oeste dos EUA)

AWS STSNo momento, o oferece suporte a VPC endpoints nas seguintes regiões:

- Leste dos EUA (Norte da Virgínia)
- Leste dos EUA (Ohio)
- Oeste dos EUA (N. da Califórnia)
- Oeste dos EUA (Oregon)
- África (Cidade do Cabo)
- Ásia-Pacífico (Hong Kong)
- Ásia-Pacífico (Hyderabad)

- Ásia-Pacífico (Jacarta)
- Ásia-Pacífico (Melbourne)
- Ásia-Pacífico (Mumbai)
- Ásia-Pacífico (Osaka)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)
- Ásia-Pacífico (Tóquio)
- Canadá (Central)
- Oeste do Canadá (Calgary)
- China (Pequim)
- China (Ningxia)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Milão)
- Europa (Paris)
- Europa (Espanha)
- Europa (Estocolmo)
- Europa (Zurique)
- Israel (Tel Aviv)
- Oriente Médio (Bahrein)
- Oriente Médio (Emirados Árabes Unidos)
- América do Sul (São Paulo)
- AWS GovCloud (Leste dos EUA)
- AWS GovCloud (Oeste dos EUA)

Criar um VPC endpoint para o IAM

Para começar a usar o IAM com sua VPC, crie um endpoint de VPC de interface para o IAM. Para obter mais informações, consulte Acessar um serviço da AWS por meio de um endpoint da VPC de interface no Guia do usuário do Amazon VPC.

Os endpoints da VPC para o IAM podem ser criados somente na região em que o ambiente de gerenciamento do IAM está localizado. Nas Regiões da AWS comerciais, o ambiente de gerenciamento do IAM está localizado na região Leste dos EUA (Norte da Virgínia) (us-east-1). O nome do serviço de endpoint da VPC de interface do AWS PrivateLink para o IAM é `com.amazonaws.iam`. Para obter uma lista de Regiões da AWS que oferecem suporte a endpoints de VPC para o IAM, consulte Disponibilidade de endpoint da VPC.

Se a sua VPC estiver em uma região diferente da região do ambiente de gerenciamento do IAM, será necessário usar o AWS Transit Gateway para permitir acesso ao endpoint da VPC de interface do IAM de outra região.

Para acessar um endpoint de VPC de interface do IAM a partir de uma VPC em uma região diferente usando o AWS Transit Gateway

1. Crie um gateway de trânsito ou use um gateway de trânsito existente para interconectar suas nuvens privadas virtuais (VPCs). É necessário um gateway de trânsito para cada região. Para obter mais informações, consulte Criar um gateway de trânsito no Guia do AWS Transit Gateway.
2. Crie anexos de VPC de gateway de trânsito para conectar cada VPC ao gateway de trânsito. Para obter mais informações, consulte Criar um anexo ao gateway de trânsito para uma VPC no Guia do AWS Transit Gateway.
3. Crie um anexo de emparelhamento de VPC de gateway de trânsito para rotear o tráfego entre VPCs emparelhadas. Para saber mais, consulte Criar um anexo de emparelhamento, no Guia do AWS Transit Gateway.

> **Note**
>
> As conexões de emparelhamento de VPC também podem rotear o tráfego entre VPCs emparelhadas, mas esse método não se adapta bem a muitas VPCs. Em vez do emparelhamento de VPC, recomendamos anexos de emparelhamento AWS Transit Gateway que melhoram o gerenciamento da VPC e da rede on-premises por meio de um hub central

escalável. Para obter mais informações sobre conexões de emparelhamento da VPC, consulte Trabalhar com conexões de emparelhamento da VPC, no Guia de emparelhamento do Amazon VPC.

Criar um VPC endpoint para o AWS STS

Para começar a usar o AWS STS com sua VPC, crie um VPC endpoint de interface para o AWS STS. Para obter mais informações, consulte Acessar um serviço da AWS por meio de um endpoint da VPC de interface no Guia do usuário do Amazon VPC.

Depois de criar o VPC endpoint, você deve usar o endpoint regional correspondente para enviar suas solicitações do AWS STS. O AWS STS recomenda que você use ambos os métodos `setRegion` e `setEndpoint` para fazer chamadas para um endpoint regional. Você pode usar o método `setRegion` sozinho para regiões habilitadas manualmente, como Ásia-Pacífico (Hong Kong). Nesse caso, as chamadas são direcionadas para o endpoint regional do STS. Para saber como habilitar manualmente uma região, consulte Gerenciar regiões da AWS no Referência geral da AWS. Se você usar o método `setRegion` sozinho para regiões habilitadas por padrão, as chamadas são direcionadas para o endpoint global de `https://sts.amazonaws.com` .

Quando você usa endpoints regionais, o AWS STS chama outros serviços da AWS usando VPC endpoints públicos ou privados de interface, o que estiver em uso. Por exemplo, suponha que você tenha criado uma interface VPC endpoint para o AWS STS e já solicitou as credenciais temporárias do AWS STS de recursos que estão localizados na VPC. Nesse caso, essas credenciais começam fluindo por meio da interface VPC endpoint por padrão. Para obter mais informações sobre como fazer solicitações regionais usando o AWS STS, consulte Gerenciar o AWS STS em uma Região da AWS.

Criar um endpoint da VPC para descoberta do OIDC do AWS STS

A descoberta do OIDC do AWS Security Token Service (AWS STS) é um par de endpoints públicos que fornecem os metadados e as chaves de verificação necessárias para validar os JSON Web Tokens (JWTs) emitidos pela federação de identidade externa.

Quando você habilita a federação de identidade externa, o AWS STS publica um URL de emissor específico da conta. Esse URL hospeda dois endpoints de descoberta do OpenID Connect (OIDC). As partes confiáveis buscam esses endpoints para obter os metadados e as chaves públicas de que precisam para verificar os JWTs que o GetWebIdentityToken emite:

- `{issuer_url}/.well-known/openid-configuration`: o documento de descoberta do OIDC
- `{issuer_url}/.well-known/jwks.json`: o JSON Web Key Set (JWKS) que contém as chaves de verificação de assinatura de token

Por padrão, acessar estes endpoints a partir da sua VPC requer um gateway da internet, dispositivo NAT, conexão VPN ou conexão. Em vez disso, para manter este tráfego na rede da AWS, crie um endpoint da VPC de interface para o endpoint de descoberta do STS OIDC.

> **Note**
>
> O endpoint de descoberta do STS OIDC atende somente aos dois endpoints listados acima. Ele não atende nenhuma operação de API do AWS STS. Para chamar `GetWebIdentityToken`, `AssumeRole` ou qualquer outra operação do AWS STS de forma privada, use o serviço de endpoint `com.amazonaws.`*`region`*`.sts`. Consulte Criar um VPC endpoint para o AWS STS.

Como escolher o serviço de endpoint correto

A federação de identidade externa usa dois caminhos de rede distintos. A maioria das workloads que cria e verifica tokens precisa dos dois serviços de endpoint.

Tarefa	Serviço de endpoint	Autenticação
Criar um JWT (`sts:GetWebIdentityToken`)	`com.amazonaws.`*`region`*`.sts`	AWSSigV4 da
Leia… `/.well-known/openid-configuration`	`com.amazonaws.`*`region`*`.sts-oidc`	Nenhum (unauthenticated)
Leia… `/.well-known/jwks.json`	`com.amazonaws.`*`region`*`.sts-oidc`	Nenhum (unauthenticated)

Você não precisa do endpoint de descoberta do STS OIDC se a única parte que verifica seus tokens for um serviço externo fora da AWS. Este serviço busca o URL do emissor pela internet pública e o tráfego não entra na sua VPC.

Considerações

Considere o seguinte antes de criar um endpoint da VPC de interface para a descoberta do STS OIDC.

- As Políticas de endpoint da VPC não são compatíveis. Ambas as APIs de descoberta fornecem dados disponíveis publicamente e o endpoint de descoberta STS OIDC não impõe políticas de endpoint da VPC. Use grupos de segurança e posicionamento de sub-rede para controlar quais recursos podem acessar o endpoint.
- Somente dois caminhos são disponibilizado. O endpoint aceita solicitações HTTPS `GET` somente para `/.well-known/openid-configuration` e `/.well-known/jwks.json`. Todos os outros caminhos e métodos retornam um erro.

Criar um endpoint da VPC de interface para a descoberta do STS OIDC

É possível criar um endpoint da VPC para o endpoint de descoberta do STS OIDC usando a console do Amazon Virtual Private Cloud ou a AWS Command Line Interface. Para obter mais informações, consulte Acessar um serviço da AWS por meio de um endpoint da VPC de interface no Guia do usuário do Amazon VPC.

Crie um endpoint da VPC para a descoberta do STS OIDC usando o seguinte nome de serviço:

- `com.amazonaws.`*`region`*`.sts-oidc`

Se habilitar o DNS privado para o endpoint, poderá enviar solicitações para `sts-oidc.`*`region`*`.amazonaws.com` e elas serão resolvidas para as interfaces de rede do endpoint na sua VPC.

Envio de solicitações por meio do endpoint

Quando uma parte confiável em sua VPC precisa verificar um JWT, ela busca o documento de descoberta do OIDC e o JWKS do URL do emissor. O nome do host do emissor (por exemplo, *`uuid`*`.tokens.sts.global.api.aws`) usa um domínio diferente do endpoint (`sts-oidc.`*`region`*`.amazonaws.com`), portanto, não é resolvido para o endpoint da VPC por padrão.

Para rotear solicitações do nome de host do emissor pelo endpoint, crie uma zona hospedada privada do Amazon Route 53 para o nome de host do emissor e adicione um registro de alias que aponte para o endpoint da VPC. Depois de criar a zona hospedada, os recursos na VPC resolvem

o nome de host do emissor para as interfaces de rede do endpoint. As bibliotecas padrão do OIDC podem então buscar os documentos de descoberta sem alterações no código. Para obter mais informações sobre como criar uma zona hospedada privada, consulte Trabalhar com zonas hospedadas privadas no Guia do desenvolvedor do Amazon Route 53.

> **Note**
>
> O serviço de endpoint é regional, mas um único endpoint atende às solicitações de descoberta para qualquer UUID do emissor na partição. Crie endpoints nas regiões em que seus verificadores são executados; você não precisa de um por conta de emissor.

Controle de acesso ao AWS STS com as políticas de endpoint da VPC

Ao criar um endpoint de interface da VPC para AWS Security Token Service (AWS STS), é possível anexar uma política de endpoint. A política controla quais entidades principais podem usar o endpoint e quais ações do AWS STS elas podem realizar. Se você não anexar uma política, o endpoint usará a política padrão que permite acesso irrestrito a todas as ações do AWS STS para todas as entidades principais.

As políticas de endpoint da VPC não concedem permissões por conta própria. Elas atuam como um limite adicional que funciona junto com outras políticas. Tanto a política do endpoint quanto as políticas aplicáveis do chamador precisam permitir uma solicitação para que ela tenha êxito.

Para obter ter mais informações sobre as políticas de endpoint da VPC, consulte Controlar o acesso aos endpoints da VPC usando políticas de endpoint no Guia do usuário da Amazon VPC.

Tópicos

- Política de endpoint da VPC padrão
- Considerações importantes sobre as políticas de endpoint da VPC do AWS STS
- Chaves de condição disponíveis para políticas de endpoint da VPC do AWS STS
- Exemplo: permitir todas as ações do AWS STS para sua organização
- Exemplo: permitir todas as ações do AWS STS para contas específicas
- Exemplo: restringir a ações específicas do AWS STS
- Exemplo: negar acesso de pessoas não pertencentes à organização e, ao mesmo tempo, permitir a federação

Política de endpoint da VPC padrão

Se você não anexar uma política personalizada ao criar o endpoint, a AWS anexará a política padrão a seguir. Essa política permite acesso irrestrito ao endpoint.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

Para limitar o acesso ao endpoint, anexe uma política de endpoint personalizada.

Considerações importantes sobre as políticas de endpoint da VPC do AWS STS

O AWS STS lida com solicitações de dois tipos fundamentalmente diferentes de chamadores. Sua política de endpoint da VPC precisa considerar os dois tipos para evitar o bloqueio involuntário de solicitações legítimas.

Entidades principais da AWS autenticadas

Usuários do IAM e perfis do IAM que assinam as solicitações com o AWS Signature Version 4 (SigV4). Esses chamadores têm chaves de condição padronizadas, como `aws:PrincipalOrgID`, `aws:PrincipalAccount` e `aws:PrincipalArn`, disponíveis no contexto da solicitação.

Chamadores federados

Entidades principais SAML 2.0 e OpenID Connect (OIDC) que chamam `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity`. Esses chamadores se autenticam com asserções SAML ou JSON Web Tokens (JWTs), não com assinaturas SigV4. Como eles não têm nenhuma identidade da AWS no momento da solicitação, o contexto da solicitação não inclui as chaves de condição baseadas nas entidades principais, como `aws:PrincipalOrgID`, `aws:PrincipalAccount` e `aws:PrincipalArn`.

> **Important**
>
> Se sua política de endpoint da VPC depende apenas de `aws:PrincipalOrgID` para permitir o acesso, as chamadas federadas `AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity` serão negadas implicitamente porque não há chave de condição para as entidades principais que não são da AWS.

Como o AWS STS avalia as políticas de endpoint da VPC para chamadores federados

Quando um chamador federado invoca `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity` por meio de um endpoint da VPC, o seguinte se aplica:

- O chamador não é uma entidade principal da AWS. As chaves de condição, como `aws:PrincipalOrgID`, `aws:PrincipalAccount` e `aws:PrincipalArn`, não estão disponíveis no contexto da solicitação.
- Os chamadores federados não têm a chave de condição `aws:PrincipalIsAWSService` no contexto da solicitação.
- A função que está sendo assumida é um recurso da AWS. As chaves de condição baseadas em recursos, como `aws:ResourceOrgID` e `aws:ResourceAccount`, estão disponíveis e se referem à função de destino.
- A política de confiança da função continua sendo a principal porta de autorização para o acesso federado. A política de endpoint da VPC fornece um limite adicional de rede.

Recomendamos o uso de `aws:ResourceOrgID` ou `aws:ResourceAccount` ao escrever as instruções da política de endpoint da VPC que devem ser aplicadas aos chamadores federados porque esses chamadores não têm chaves de condição disponíveis baseadas nas entidades principais.

Chaves de condição disponíveis para políticas de endpoint da VPC do AWS STS

A tabela a seguir mostra as chaves de condição comumente usadas e sua disponibilidade no contexto da solicitação para cada tipo de chamador ao fazer solicitações por meio de um endpoint da VPC do AWS STS. Estão disponíveis chaves de condição adicionais, além das listadas aqui.

Disponibilidade da chave de condição por tipo de chamador

Chave de condição	Entidades principais da AWS autenticadas	Chamadores federados	Descrição
`aws:PrincipalOrgID`	Sim	Não	A ID da organização da entidade principal que faz a chamada
`aws:PrincipalAccount`	Sim	Não	A ID da conta da entidade principal que faz a chamada
`aws:PrincipalArn`	Sim	Não	O ARN da entidade principal que faz a chamada
`aws:PrincipalIsAWSService`	Sim (avalia como falso)	Não (a chave está ausente)	Se o chamador é uma entidade principal de um serviço da AWS
`aws:ResourceOrgID`	Sim	Sim	A ID da organização da conta que possui o recurso solicitado
`aws:ResourceAccount`	Sim	Sim	A ID da conta que possui o recurso solicitado

Note

No caso de entidades principais da AWS autenticadas (usuários e perfis do IAM), o `aws:PrincipalIsAWSService` está presente no contexto da solicitação e avalia como falso. Para chamadores federados, essa chave está totalmente ausente do contexto da solicitação. Uma condição que verifica `"Bool": {"aws:PrincipalIsAWSService": "false"}` não corresponde aos chamadores federados porque a chave não está presente.

Exemplo: permitir todas as ações do AWS STS para sua organização

A política de endpoint a seguir restringe o endpoint da VPC do AWS STS à sua organização e, ao mesmo tempo, oferece suporte ao acesso federado. Ela permite todas as ações do AWS STS para as entidades principais autenticadas em sua organização e, separadamente, permite que chamadores federados assumam funções em sua organização. Os chamadores federados (`AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity`) precisam de uma instrução separada porque não têm uma `aws:PrincipalOrgID` no contexto da solicitação.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrganizationPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
    ]
```

```
}
```

A primeira instrução usa `"Principal": {"AWS": "*"}` com `aws:PrincipalOrgID` para permitir entidades principais da AWS autenticadas de sua organização. A segunda instrução usa `"Principal": "*"` para corresponder aos chamadores federados e restringe a função de destino à sua organização mediante o uso de `aws:ResourceOrgID`. A política de confiança da função continua sendo o controle principal para definir as funções que as identidades federadas podem assumir.

Para obter mais informações sobre a implementação de controles de perímetro de rede, consulte Como criar um perímetro de dados na AWS e os exemplos de políticas de perímetro de dados no site do GitHub.

Exemplo: permitir todas as ações do AWS STS para contas específicas

A política de endpoint a seguir permite todas as ações do AWS STS para entidades principais em contas específicas. Use essa política quando suas contas não fizerem parte de uma organização. Os chamadores federados são permitidos em uma instrução separada porque não têm uma `aws:PrincipalAccount` no contexto da solicitação.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSpecificAccountPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        },
        {
```

```
            "Sid": "AllowFederatedAssumeRole",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:AssumeRoleWithWebIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": [
                        "111122223333",
                        "444455556666"
                    ]
                }
            }
        }
    ]
}
```

A primeira instrução usa `"Principal": {"AWS": "*"}` com `aws:PrincipalAccount` para permitir entidades principais da AWS autenticadas das contas especificadas. A segunda instrução usa `"Principal": "*"` para fazer a correspondência dos chamadores federados e restringe a função de destino às mesmas contas mediante o uso de `aws:ResourceAccount`. A política de confiança da função continua sendo o controle principal para definir as funções que as identidades federadas podem assumir.

Exemplo: restringir a ações específicas do AWS STS

A política de endpoint a seguir permite somente `AssumeRole` para as entidades principais autenticadas e `AssumeRoleWithSAML` para os chamadores federados, ambos com escopo definido para funções em sua organização.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAssumeRoleByOrgPrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
```

```
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        },
        {
            "Sid": "AllowSAMLFederation",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "sts:AssumeRoleWithSAML",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

Essa política bloqueia outras ações do AWS STS, como `GetCallerIdentity`, `GetSessionToken` e `AssumeRoleWithWebIdentity`, por meio desse endpoint. Ajuste os elementos da `Action` para atender às suas necessidades.

Exemplo: negar acesso de pessoas não pertencentes à organização e, ao mesmo tempo, permitir a federação

A política de endpoint a seguir usa uma negação explícita para bloquear as entidades principais autenticadas de fora da sua organização, enquanto preserva o acesso para chamadores federados. Essa abordagem começa com uma permissão ampla e adiciona uma instrução de negação direcionada.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAll",
            "Effect": "Allow",
            "Principal": "*",
```

```
            "Action": "*",
            "Resource": "*"
        },
        {
            "Sid": "DenyNonOrgPrincipals",
            "Effect": "Deny",
            "Principal": {
                "AWS": "*"
            },
            "Action": "sts:*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalOrgID": "o-exampleorgid"
                }
            }
        }
    ]
}
```

A instrução de negação usa `"Principal": {"AWS": "*"}`, que a define apenas para as entidades principais da AWS autenticadas. Os chamadores federados (SAML e OIDC) não são entidades principais da AWS e não são correspondidos por esse elemento `Principal`. Portanto, a negação não se aplica a eles. Essa abordagem evita a necessidade de condições complexas `Null` ou `Bool` para criar exceções para chamadores federados.

> **Note**
>
> A primeira instrução permite todas as ações para todas as entidades principais. A negação na segunda instrução tem precedência para as entidades principais da AWS autenticadas de fora da sua organização. Os chamadores federados são permitidos pela primeira instrução e não são afetados pela negação porque o elemento `Principal` na instrução de negação não corresponde a eles.

Uso de chaves de condição da VPC para controlar o acesso federado

Os usuários federados podem assumir funções usando `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity` por meio de um endpoint da VPC. Quando essas funções são assumidas, é possível usar chaves de condição específicas da VPC na política de confiança da

função ou nas políticas de controle de recursos (RCPs) para restringir a origem dessas solicitações. Essa ação delimita a rede e permite controles de identidade na política de confiança.

Tópicos

- Como as chaves de condição da VPC funcionam no caso de solicitações federadas
- Exemplo: restringir a federação do OIDC a uma VPC específica
- Exemplo: restringir a federação do SAML a um endpoint da VPC específico
- Exemplo: restringir a federação do OIDC por intervalo de IP privado em uma VPC
- Exemplo: restringir a federação de SAML a várias VPCs
- Exemplo: combinar condições específicas da VPC e do provedor
- Exemplo: usar uma RCP para restringir o acesso federado a uma VPC específica

Como as chaves de condição da VPC funcionam no caso de solicitações federadas

As chaves de condição disponíveis no contexto da solicitação dependem do caminho de rede da solicitação federada.

Disponibilidade da chave de condição da VPC de acordo com o caminho de rede

Chave de condição	Por meio do endpoint da VPC	Pela internet pública	Descrição
`aws:SourceVpc`	Sim	Não	ID da VPC que a solicitação percorre
`aws:SourceVpcArn`	Sim	Não	ARN da VPC que a solicitação percorre
`aws:SourceVpce`	Sim	Não	ID do endpoint da VPC que a solicitação percorre
`aws:VpcSourceIp`	Sim	Não	Endereço IP privado do chamador na VPC
`aws:SourceIp`	Não	Sim	Endereço IP público do chamador

Se sua política de confiança usa `aws:SourceVpc`, `aws:SourceVpcArn` ou `aws:SourceVpce` em uma instrução `Allow` e a solicitação não é realizada por meio de um endpoint da VPC, a condição não é considerada correspondente e a AWS nega implicitamente a solicitação. Isso exige que as solicitações federadas percorram efetivamente o endpoint da VPC para que tenham êxito.

> **Important**
>
> No caso de solicitações feitas por meio de um endpoint da VPC, `aws:SourceIp` não é preenchido. Em vez disso, use `aws:VpcSourceIp` para restrições com base em IP.

Recomendamos o uso de `aws:SourceVpcArn`, em vez de `aws:SourceVpc`, nas políticas de confiança. As IDs da VPC são exclusivas regionalmente, mas não globalmente, ao passo que `aws:SourceVpcArn` inclui a região e a conta, fornecendo uma identificação exclusiva global da VPC.

> **Note**
>
> O valor de audiência (`aud`) nos exemplos de OIDC a seguir usa `sts.amazonaws.com`, que é o padrão para o Amazon EKS. Para outros provedores de OIDC, substitua esse valor pelo valor do destinatário (audience value) configurado para sua aplicação (por exemplo, ID do cliente ou URI da aplicação).

Exemplo: restringir a federação do OIDC a uma VPC específica

A política de confiança a seguir permite que um provedor de OIDC assuma a função somente quando a solicitação tem como origem uma VPC específica por meio de um endpoint da VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromVpc",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
```

```
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

A AWS nega implicitamente as solicitações provenientes de fora da VPC especificada porque a condição `aws:SourceVpcArn` não é correspondente. Nenhuma instrução de negação explícita é necessária.

Exemplo: restringir a federação do SAML a um endpoint da VPC específico

A política de confiança a seguir permite que um provedor de SAML assuma a função somente quando a solicitação tem como origem um endpoint da VPC específico.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromVpce",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpce": "vpce-0abcdef1234567890"
                }
            }
        }
    ]
}
```

O uso de `aws:SourceVpce`, em vez de `aws:SourceVpc`, fornece um controle mais detalhado quando há vários endpoints da VPC na mesma VPC e você deseja restringir o acesso a um endpoint específico.

Exemplo: restringir a federação do OIDC por intervalo de IP privado em uma VPC

A política de confiança a seguir restringe a federação às solicitações originadas de uma sub-rede específica dentro da VPC. Use `aws:VpcSourceIp` para restrições baseadas em IP em solicitações realizadas por meio de um endpoint da VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCFromSubnet",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "IpAddress": {
                    "aws:VpcSourceIp": "10.0.1.0/24"
                }
            }
        }
    ]
}
```

> **Important**
>
> Não use `aws:SourceIp` com intervalos de IP privados para solicitações por meio de um endpoint da VPC. A chave `aws:SourceIp` não está disponível para solicitações realizadas por um endpoint da VPC. Use `aws:VpcSourceIp` em vez disso.

Exemplo: restringir a federação de SAML a várias VPCs

A política de confiança a seguir permite que um provedor de SAML assuma a função de qualquer uma das várias VPCs. Isso é útil quando sua organização tem várias VPCs em diferentes ambientes ou regiões que precisam de acesso federado à mesma função.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSAMLFromMultipleVpcs",
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:saml-provider/ExampleProvider"
            },
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "SAML:aud": "https://signin.aws.amazon.com/saml",
                    "aws:SourceVpcArn": [
                        "arn:aws:ec2:us-west-2:111122223333:vpc/vpc-111aaa11",
                        "arn:aws:ec2:us-east-1:111122223333:vpc/vpc-222bbb22"
                    ]
                }
            }
        }
    ]
}
```

Exemplo: combinar condições específicas da VPC e do provedor

Você pode combinar chaves de condição da VPC com condições específicas do provedor, como audience (`aud`) e subject (`sub`), para um controle de acesso mais detalhado. A política de confiança a seguir restringe a um usuário federado específico a execução de uma função e exige que a solicitação tenha como origem uma VPC específica.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOIDCUserFromVpc",
            "Effect": "Allow",
```

```
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "idp.example.com:aud": "sts.amazonaws.com",
                    "idp.example.com:sub": "user123",
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                }
            }
        }
    ]
}
```

A política combina controles de identidade (restringindo a um usuário federado específico) com controles de rede (restringindo a uma VPC específica), proporcionando uma defesa profunda.

Exemplo: usar uma RCP para restringir o acesso federado a uma VPC específica

A política de controle de recursos (RCP) a seguir nega as solicitações `AssumeRoleWithSAML` e `AssumeRoleWithWebIdentity`, a menos que tenham sido originadas por uma VPC específica. Ao contrário das políticas de confiança que se aplicam por função, uma política de controle de recursos (RCP) se aplica a todas as funções nas contas de destino. Isso fornece um limite de rede centralizado.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyFederatedAccessOutsideVpc",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "sts:AssumeRoleWithWebIdentity",
                "sts:AssumeRoleWithSAML"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
```

```
                    "aws:SourceVpcArn": "arn:aws:ec2:us-west-2:111122223333:vpc/
vpc-111bbb22"
                },
                "Null": {
                    "aws:SourceVpcArn": "false"
                }
            }
        }
    ]
}
```

A condição `Null` garante que a negação só se aplique quando `aws:SourceVpcArn` estiver presente no contexto da solicitação (ou seja, a solicitação teve como origem um endpoint da VPC, mas da VPC errada). Sem isso, as solicitações pela internet pública (em que `aws:SourceVpcArn` está ausente) também seriam negadas por `StringNotEquals`. Se a verificação `Null` for removida, a negação também se aplicará às solicitações que não percorram nenhum endpoint da VPC, inclusive as solicitações pela internet pública. Recomendamos testar minuciosamente antes de remover a verificação `Null` nos ambientes de produção.

AWSServiços da que funcionam com o IAM

Os serviços da AWS listados abaixo estão agrupados em ordem alfabética e incluem informações sobre a compatibilidade deles com atributos do IAM:

- Serviço: você pode selecionar o nome de um serviço para visualizar a documentação da AWS sobre a autorização e o acesso do IAM para esse serviço.
- Ações: você pode especificar ações individuais em uma política. Se o serviço não é compatível com esse atributo, então Todas as ações está selecionado no editor visual. Em um documento de política JSON, use `*` no elemento `Action`. Para obter uma lista de ações em cada serviço, consulte Ações, recursos e chaves de condição para serviços da AWS.
- Permissões no nível do recurso: você pode usar ARNs para especificar recursos individuais na política. Se o serviço não é compatível com esse atributo, então Todos os recursos está selecionado no editor visual de política. Em um documento de política JSON, use `*` no elemento `Resource`. Algumas ações, como ações `List*`, não são compatíveis com a especificação de um ARN, pois elas são projetadas para retornar vários recursos. Se um serviço é compatível com essa funcionalidade para alguns atributos e não outros, isso é indicado por Partial (Parcial) na tabela. Consulte a documentação do serviço para obter mais informações.

- Políticas baseadas em recursos: você pode anexar políticas baseadas em recursos a um recurso do serviço. As políticas baseadas em recursos incluem um elemento `Principal` para especificar quais identidades do IAM podem acessar esse recurso. Para obter mais informações, consulte Políticas baseadas em identidade e em recurso.
- ABAC (autorização baseada em tags): para controlar o acesso baseado em tags, você fornece informações sobre as tags no elemento de condição de uma política usando as chaves de condição `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Se um serviço for compatível com as três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço oferecer suporte às três chaves de condição somente para alguns tipos de recursos, o valor será Partial (Parcial).

  Para obter mais informações sobre como definir permissões com base em atributos como tags, consulte Definir permissões com base em atributos com autorização ABAC. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Use attribute-based access control (ABAC) (Usar controle de acesso por atributo (ABAC)).
- Credenciais temporárias — É possível usar credenciais de curto prazo obtidas ao fazer login usando o IAM Identity Center, o gerenciador de acesso à conta, alternar perfis no console, ou ainda credenciais que você gera usando o AWS STS na AWS CLI ou a API AWS. É possível acessar serviços com um valor Não somente ao usar suas credenciais de longo prazo de usuário do IAM. Isso inclui um nome de usuário e senha ou as chaves de acesso de usuário. Para obter mais informações, consulte Credenciais de segurança temporárias no IAM.
- Perfis vinculados ao serviço: um perfil vinculado ao serviço é um tipo especial de perfil de serviço que concede permissão ao serviço para acessar recursos em outros serviços para você. Escolha o link Sim ou Parcial para consultar a documentação de serviços que são compatíveis com esses perfis. Essa coluna não indica se o serviço usa perfis de serviço padrão. Para obter mais informações, consulte Perfis vinculados ao serviço.
- Mais informações – Se um serviço não é totalmente compatível com um atributo, você pode revisar as notas de rodapé de uma entrada para visualizar as restrições e links de informações relacionadas.

Serviços compatíveis com o IAM

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Gerenciador de acesso à conta	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Gerenciamento de contas	Yes (Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Recomendações de ações	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Activate Console	Sim	No (Não	Não	No (Não	Sim	Não
Operações de IA da Amazon	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Amplify Admin	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Amplify	Yes (Sim	Yes (Sim	Não	Parcial	Sim	Não
AWS Amplify UI Builder	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
APIs do Apache Kafka para clusters do Amazon MSK	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon API Gateway	Sim	Yes (Sim	Sim	No (Não	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon API Gateway Management	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon API Gateway Management V2	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS App Studio	Sim	No (Não)	Não	No (Não)	Sim	No (Não
AWS App2Container	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS AppConfig	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS AppFabric	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon AppFlow	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon AppIntegrations	Sim	Yes (Sim	No (Não)	Yes (Sim	Yes (Sim	Sim
Application Auto Scaling	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Yes (Sim
AWS Application Discovery Arsenal	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Application Discovery Service	Sim	No (Não	Não	No (Não	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Transform MGN	Yes (Sim	Yes (Sim	No (Não)	Yes (Sim	Yes (Sim	Sim
Amazon Application Recovery Controller – Mudança de zona	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Application Transformation Service	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS App Mesh	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim)	Sim
AWS App Mesh versão prévia	Sim	Yes (Sim	No (Não)	No (Não	Yes (Sim	Sim
AWS App Runner	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon AppStream 2.0	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS AppSync	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Troca de região do Amazon ARC	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Artifact	Yes (Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Athena	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim)	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Audit Manager	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Aurora DSQL	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim)	Sim
AWS Auto Scaling	Yes (Sim	No (Não	Não	No (Não	Yes (Sim	Sim
AWS Intercâmbio de dados B2B	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Backup	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Sim
AWS Backup Gateway	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Backup Pesquisa do	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Backup Armazenamento do	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Batch	Sim	Parcial	Não	Yes (Sim	Yes (Sim	Sim
Amazon Bedrock	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Bedrock Agentcore	Sim	Yes (Sim	Não	Parcial	Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Gerenciamento de Faturamento e Custos da AWS	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Painéis de Gerenciamento de Faturamento e Custos da AWS	Yes (Sim	No (Não	Não	No (Não	Sim	Não
Gerenciamento de Faturamento e Custos da AWS Exportações de dados	Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim	Não
Gerenciamento de Faturamento e Custos da AWS Calculadora de preços do	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Gerenciamento de Faturamento e Custos da AWS Ações recomendadas do	Sim	No (Não	Não	No (Não)	Sim	No (Não
AWS Billing Conductor	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Braket	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Budget Service	Sim	Yes (Sim	No (Não)	Yes (Sim	Yes (Sim	Não
Gerenciador de certificados da AWS (ACM)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Q Developer em aplicações de chat	Sim	Yes (Sim	No (Não)	Yes (Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Chime	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Clean Rooms	Yes (Sim	Yes (Sim	No (Não	Yes (Sim)	Yes (Sim	Não
AWS Clean Rooms ML	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Client VPN	Yes (Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim
AWS Cloud9	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Sim
Nuvem AWS API de controle do	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Cloud Directory	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
CloudFormation	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon CloudFront	Sim	Yes (Sim	Não	Parcial	Sim	Sim
KeyValueStore do Amazon CloudFront	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS CloudHSM	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Cloud Map	Yes (Sim)	Yes (Sim)	No (Não	Yes (Sim	Yes (Sim	Não
Amazon CloudSearch	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS CloudShell	Yes (Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS CloudTrail	Yes (Sim)	Yes (Sim	Parcial (Informaçõ es)	Sim	Yes (Sim	Sim
AWS CloudTrail Dados do	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon CloudWatch	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Parcial (Informaçõ es)
Amazon CloudWatch Application Insights	Sim	No (Não	Não	No (Não	Sim	Não
Amazon CloudWatch Application Signals	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Monitor de Internet do Amazon CloudWatch	Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim	Não
Amazon CloudWatch Logs	Sim	Yes (Sim	Sim	Parcial	Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon CloudWatch Network Synthetic Monitor	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon CloudWatch Observability Access Manager	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Serviço de administração de observabilidade do Amazon CloudWatch	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon CloudWatch RUM	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon CloudWatch Synthetics	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS CodeArtifact	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	No (Não
AWS CodeBuild	Yes (Sim	Yes (Sim	Sim (Informações)	Parcial (Informações)	Sim	Não
Amazon CodeCatalyst	Sim	Yes (Sim	No (Não)	Yes (Sim	Yes (Sim	Sim
AWS CodeCommit	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS CodeConnections	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS CodeDeploy	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS CodeDeploy Serviço de comandos de host seguro do	Sim	No (Não	Não	No (Não	Sim	Não
Amazon CodeGuru Profiler	Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim)	Sim
Amazon CodeGuru Reviewer	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon CodeGuru Security	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS CodePipeline	Sim	Parcial	Não	Yes (Sim	Yes (Sim	No (Não
AWS CodeStar	Sim	Parcial	Não	Yes (Sim	Yes (Sim	Não
AWS CodeStar Conexões do	Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim	Sim
AWS CodeStar Notificações do	Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim)	Sim
Amazon CodeWhisperer	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Cognito	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Cognito Sync	Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim
Grupos de usuários do Amazon Cognito	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Comprehend	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Comprehend Medical	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Compute Optimizer	Sim	No (Não	Não	No (Não	Yes (Sim	Sim
AWS Config	Yes (Sim	Parcial (Informações)	Não	Yes (Sim	Yes (Sim	Yes (Sim
Amazon Connect Customer	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Yes (Sim
Casos do Amazon Connect Customer	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
Perfis de clientes do Amazon Connect Customer	Yes (Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim)	Yes (Sim
Campanhas externas do Amazon Connect Customer	Yes (Sim	Yes (Sim)	No (Não	Yes (Sim)	Yes (Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
ID de voz do Amazon Connect Customer	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
Amazon Connect Health	Yes (Sim	Yes (Sim	Não	Parcial	Sim	No (Não
AWS Console Mobile Application	Yes (Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Faturamento Consolidado da	Sim	No (Não	Não	No (Não	Sim	Não
AWS Catálogo de controle da	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Control Tower	Yes (Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Cost and Usage Report	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Cost Explorer	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
Hub de Otimização de Custos da AWS	Sim	No (Não	Não	No (Não	Sim	Não
AWS Serviço de verificação de clientes da	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Database Migration Service	Yes (Sim	Yes (Sim	Não (Informações)	Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Database Query Metadata Service	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Data Exchange	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Data Lifecycle Manager	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Data Pipeline	Yes (Sim	Yes (Sim	Não	Parcial	Sim	No (Não
AWS DataSync	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon DataZone	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Deadline Cloud	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS DeepRacer	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Detective	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Device Farm	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon DevOps Guru	Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Ferramentas de diagnóstico	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim)	✓ Yes (Sim	✗ No (Não
AWS Direct Connect	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Sim	✓ Yes (Sim	✓ Sim
AWS Directory Service	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
AWS Dados do Directory Service	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon DocumentDB Elastic Clusters	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim)	✓ Yes (Sim	✓ Sim
Amazon DynamoDB Accelerator (DAX)	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Yes (Sim	✓ Sim
Amazon DynamoDB	✓ Sim	✓ Yes (Sim	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon Elastic Compute Cloud (Amazon EC2)	✓ Sim	⚠ Parcial	✗ Não	✓ Sim	✓ Yes (Sim	⚠ Parcial (Informações)
Amazon EC2 Auto Scaling	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
EC2 Image Builder	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon EC2 Instance Connect	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Elastic Beanstalk	Sim	Parcial	Não	Sim	Yes (Sim	Sim
Amazon Elastic Block Store (Amazon EBS)	Sim	Parcial	Não	Yes (Sim	Yes (Sim	Não
Amazon Elastic Container Registry (Amazon ECR)	Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Sim
Amazon Elastic Container Registry Público (Amazon ECR Público)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Elastic Container Service (Amazon ECS)	Sim	Parcial (Informações)	Não	Yes (Sim	Yes (Sim	Sim
Recuperação de desastres do AWS Elastic	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Elastic File System (Amazon EFS)	Sim	Yes (Sim	Sim	Parcial	Sim	Sim
Amazon Elastic Kubernetes Service (Amazon EKS)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Elastic Kubernetes Service (Amazon EKS) Auth	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Elastic Load Balancing	Sim	Parcial	Não	Parcial	Sim	Yes (Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Elastic VMware Service	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon ElastiCache	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Elemental Appliances and Software Activation Service	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Dispositivos e software do Elemental	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Elemental MediaConnect	Yes (Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim
AWS Elemental MediaConvert	Yes (Sim	Yes (Sim	No (Não	Sim	Yes (Sim	No (Não
AWS Elemental MediaLive	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Elemental MediaPackage	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Parcial (Informações)
AWS Elemental MediaPackage V2	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Elemental MediaPackage VOD do	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Parcial (Informações)

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Elemental MediaTailor	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Elemental Support Cases	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Elemental Support Content	Sim	No (Não	Não	No (Não)	Sim	Não
Amazon EMR	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon EMR no EKS	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon EMR Sem Servidor	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS End User Messaging SMS e Voice V2	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS End User Messaging Social	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Entity Resolution	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Não
Amazon EventBridge	Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Não
Amazon EventBridge Pipes	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Agendador do Amazon EventBridge	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Esquemas do Amazon EventBridge	✓ Sim	✓ Yes (Sim	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS Fault Injection Service	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon FinSpace	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon FinSpace API do	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ No (Não
AWS Firewall Manager	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Sim	⚠ Parcial
Fleet Hub for AWS IoT Device Management	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon Forecast	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon Fraud Detector	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
FreeRTOS	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS Nível gratuito da	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Yes (Sim)	✗ Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon FSx	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon GameLift Servers	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon GameLift Streams	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não
AWS Global Accelerator	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
AWS Glue	✓ Yes (Sim	✓ Sim	⚠ Parcial	⚠ Parcial	✓ Sim	✗ Não
AWS Glue DataBrew	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não
AWS Ground Station	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon Ground Truth Labeling	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
Amazon GuardDuty	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
AWS Health APIs e notificações do	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
AWS HealthImaging	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS HealthLake	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS HealthOmics	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não
Centro de Identidade do AWS IAM	✓ Yes (Sim	✓ Yes (Sim	✗ Não	⚠ Parcial	✓ Sim	✓ Sim
IAM Identity Center Directory	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
IAM Identity Center Identity Store	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
Serviço OIDC do IAM Identity Center	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
AWS Identity and Access Management (IAM)	✓ Sim	✓ Yes (Sim	⚠ Parcial (Informações)	⚠ Parcial (Informações)	⚠ Parcial (Informações)	✗ Não
AWS Identity and Access Management and Access Analyzer	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Sim	⚠ Parcial
AWS Identity and Access Management Roles Anywhere	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
AWS Identity Store Auth	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Identity Sync	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Import/Export	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Inspector	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Inspector Classic	Sim	No (Não	Não	No (Não	Yes (Sim	Sim
Amazon InspectorScan	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Interactive Video Service	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Interactive Video Service Chat	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
Faturamento da AWS	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS IoT Analytics	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS IoT	Sim	Sim	⚠ Parcial (Informações)	Sim	Yes (Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS IoT Core Device Advisor	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS IoT Device Tester	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS IoT Events	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS IoT FleetWise	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS IoT Greengrass	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS IoT Greengrass V2	Sim	Yes (Sim	Não	Parcial	Sim	Não
AWS IoT Jobs DataPlane	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS IoT Serviço de integrações gerenciadas do	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS IoT SiteWise	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS IoT TwinMaker	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS IoT Wireless	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Kendra	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Kendra Intelligent Ranking	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Key Management Service (AWS KMS)	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Sim
Amazon Keyspaces (para Apache Cassandra)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Managed Service for Apache Flink	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Managed Service for Apache Flink V2	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Data Firehose	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Kinesis Data Streams	Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Não
Amazon Kinesis Video Streams	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Lake Formation	Sim	No (Não	Não	No (Não	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Lambda	Yes (Sim	Yes (Sim	Sim	Parcial (Informações)	Sim	Parcial (Informações)
AWS Launch Wizard	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Lex	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Lex V2	Sim	Yes (Sim	Sim	Yes (Sim	No (Não	Sim
AWS License Manager	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS License Manager Linux Subscriptions Manager	Sim	No (Não	Não	No (Não	Sim	Não
AWS License Manager Assinaturas de usuário da	Sim	No (Não	Não	No (Não	Yes (Sim	Sim
Amazon Lightsail	Yes (Sim	Parcial (Informações)	Não	Parcial (Informações)	Sim	Yes (Sim
Amazon Location Service	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Mapas do Amazon Location Service	Sim	Yes (Sim	No (Não	No (Não	Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Locais do Amazon Location Service	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Rotas do Amazon Location Service	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Lookout for Equipment	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Lookout for Metrics	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Machine Learning	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Macie	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Mainframe Modernization	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Mainframe Modernization Teste de aplicação	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Managed Blockchain	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Consulta ao Amazon Managed Blockchain	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Managed Grafana	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Managed Service for Prometheus	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon Managed Streaming for Apache Kafka (MSK)	✓ Sim	✓ Yes (Sim	⚠ Parcial (Informações)	✓ Sim	✓ Yes (Sim	✓ Sim
Amazon Managed Streaming for Kafka Connect	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon Managed Workflows for Apache Airflow	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Yes (Sim
AWS Marketplace	✓ Yes (Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Yes (Sim	✓ Sim
AWS Marketplace Catálogo do	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não
AWS Marketplace Commerce Analytics	✓ Sim	✗ No (Não	✗ Não	✗ Não	✗ Não	✗ Não
AWS Marketplace Serviço de implantação do	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS Marketplace Descoberta do	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ No (Não
AWS Marketplace Entitlement Service	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Marketplace Image Building Service	Sim	No (Não	Não	No (Não	Sim	No (Não
Portal de gerenciamento do AWS Marketplace	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Marketplace Metering Service	Sim	No (Não	Não	No (Não	Sim	Não
AWS Marketplace Private Marketplace	Sim	No (Não	Não	No (Não	Sim	Não
AWS Marketplace Procurement Systems Integration	Sim	No (Não	Não	No (Não	Sim	Não
AWS Marketplace Relatórios do	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Marketplace Relatórios do vendedor do	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Marketplace Vendor Insights	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon MediaImport	Sim	No (Não	Não	Não	Não	Não
Amazon MemoryDB	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Serviço de entrega de mensagens da Amazon	Sim	No (Não	Não	No (Não	Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Serviço Amazon Message Gateway	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Microservice Extractor for .NET	Sim	No (Não	Não	No (Não	Sim	Não
AWS Créditos do Programa para Aceleração da Migração para a	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Migration Hub	Yes (Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim
AWS Migration Hub Orchestrator	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Migration Hub Refactor Spaces	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Sim
AWS Migration Hub Strategy Recommendations	Sim	No (Não	Não	No (Não	Yes (Sim	Sim
Amazon Monitron	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon MQ	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Neptune	Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Aprovação Multilateral	Sim	Yes (Sim	Não	Parcial (Informações)	Sim	Não
Amazon Neptune Analytics	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Network Firewall	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Monitor de fluxo de rede	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Network Manager	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim (Informações)
AWS Network Manager Chat	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Nimble Studio	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon One Enterprise	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon OpenSearch	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon OpenSearch Ingestion	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon OpenSearch Sem Servidor	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon OpenSearch Service	Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	Sim
Oracle Database@AWS	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS OpsWorks	Yes (Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS OpsWorks Gerenciamento de configuração da	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Organizations	Yes (Sim	Yes (Sim	Sim	Yes (Sim	No (Não	Sim
AWS Outposts	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Yes (Sim
AWS Serviço de computação paralela da	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Gerenciamento de contas da Central de Parceiros da	Sim	No (Não	Não	No (Não	Sim	Não
AWS Vendas da Central de Parceiros da	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Payment Cryptography	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Payments	Sim	No (Não	Não	No (Não	Sim	Não
AWS Performance Insights	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Personalize	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Pinpoint	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Serviço de e-mail do Amazon Pinpoint	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
SMS e serviço de voz do Amazon Pinpoint	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Polly	Sim	Yes (Sim	No (Não	No (Não	Sim	No (Não
AWS Price List	Sim	No (Não	Não	No (Não	Sim	Não
AWS G privado	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
CA privada da AWS Connector for Active Directory	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
CA privada da AWS Conector para SCEP da	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Autoridade de certificação privada da AWS (CA privada da AWS)	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	No (Não
AWS PrivateLink	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Proton	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Purchase Orders Console	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Q Business	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Aplicações do Amazon Q Business Q	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Q Developer	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Q no Connect	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Quantum Ledger Database (Amazon QLDB)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
Amazon Quick	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
API Data do Amazon RDS	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Autenticação do IAM do Amazon RDS	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
AWS Lixeira da	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Redshift	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
API de dados do Amazon Redshift	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Redshift Sem Servidor	Sim	Yes (Sim	Sim (Informações)	Sim	Yes (Sim	Não
Amazon Rekognition	Sim	Yes (Sim	⚠ Parcial (Informações)	Sim	Yes (Sim	Não
Amazon Relational Database Service (Amazon RDS) (Informações)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS re:Post Privado	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Resilience Hub	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Resource Access Manager (AWS RAM)	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Explorador de recursos da AWS	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Grupos de recursos da AWS	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Sim	⚠ Parcial (Informações)	✓ Sim
AWS Resource Groups Tagging API	✓ Yes (Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
Amazon RHEL Knowledgebase Portal	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ No (Não
AWS RoboMaker	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Sim	✓ Yes (Sim	✓ Sim
Amazon Route 53	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
Domínios do Amazon Route	✓ Sim	✗ No (Não	✗ Não	✗ Não	✗ Não	✗ Não
Perfis do Amazon Route	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon Route 53 Recovery Cluster	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
Amazon Route 53 Recovery Control Config	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Route 53 Recovery Readiness	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Amazon Route 53 Resolver	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Yes (Sim
AWS RTB Fabric	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Yes (Sim
Amazon S3 Express	✓ Sim	✓ Yes (Sim	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon Glacier	✓ Sim	✓ Yes (Sim	✓ Sim	✗ No (Não	✓ Sim	⚠ Parcial
Tabelas do Amazon S	✓ Sim	✓ Yes (Sim	✓ Sim	✗ No (Não	✓ Sim	✗ Não
Amazon S3 Vectors	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
Amazon SageMaker AI	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	⚠ Parcial (Informações)
Assistente de ciência de dados do Amazon SageMaker AI	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
Recursos geoespaciais do Amazon SageMaker AI	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
Amazon SageMaker AI com MLflow	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Savings Plans	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Secrets Manager	Yes (Sim	Yes (Sim	Sim	Yes (Sim	Yes (Sim	No (Não
AWS Security Hub CSPM	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Resposta a Incidentes de Segurança da	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Security Lake	Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim
AWS Security Token Service (AWS STS)	Yes (Sim	Parcial (Informações)	Não	Sim	Parcial (Informações)	Não
AWS Serverless Application Repository	Yes (Sim	Yes (Sim	Sim	No (Não	Sim	No (Não
AWS Service Catalog	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Serviço da AWS para gerenciar os recursos de experiência do usuário do Console da AWS	Sim	No (Não	Não	No (Não	Sim	Não
Service Quotas	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Shield	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Shield Administrador de segurança da rede do	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Signer	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Fazer login	Sim	No (Não	Sim	No (Não	Sim	Não
Amazon SimpleDB	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Simple Email Service - Mail Manager	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon Simple Email Service (Amazon SES) v2	Sim	Parcial (Informações)	Sim	Yes (Sim	Parcial (Informações)	Sim
Amazon Simple Notification Service (Amazon SNS)	Sim	Yes (Sim	Yes (Sim	Yes (Sim	Yes (Sim	Não
Amazon Simple Queue Service (Amazon SQS)	Sim	Yes (Sim	Sim	Parcial	Sim	Não
Amazon Simple Storage Service (Amazon S3)	Sim	Yes (Sim	Sim	Parcial (Informações)	Sim	Parcial (Informações)

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Objeto Lambda do Amazon Simple Storage Service (Amazon S	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon Simple Storage Service (Amazon S3) no AWS Outposts	Sim	Yes (Sim	Sim	No (Não	Yes (Sim	Sim
Amazon Simple Workflow Service (Amazon SWF)	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS SimSpace Weaver	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Site-to-Site VPN	Yes (Sim	Yes (Sim	No (Não	No (Não	Yes (Sim	Sim
AWS Snowball Edge	Yes (Sim	No (Não	Não	No (Não	Sim	Não
AWS Snowball Edge Edge	Sim	No (Não	Não	No (Não	Sim	No (Não
AWS Snowball Edge Device Management	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS SQL Workbench	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Step Functions	Yes (Sim	Yes (Sim	No (Não	Sim	Yes (Sim	No (Não
AWS Storage Gateway	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Cadeia de Suprimentos AWS	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ No (Não
AWS Support	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Yes (Sim	✓ Sim
AWS Support App in Slack	✓ Yes (Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
AWS Support Console do	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
AWS Support Planos	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
AWS Sustainability	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ No (Não
AWS Systems Manager	✓ Yes (Sim	✓ Sim	⚠ Parcial	✓ Sim	✓ Yes (Sim	✓ Sim
AWS Systems Manager para SAP	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS Systems Manager GUI Connect	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ No (Não
AWS Systems Manager Incident Manager	✓ Yes (Sim	✓ Yes (Sim	✓ Sim	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
AWS Systems Manager Incident Manager Contatos do	✓ Sim	✓ Yes (Sim	✓ Sim	✗ No (Não	✓ Sim	✗ Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS Systems Manager Configuração Rápida da	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Tag Editor	Sim	No (Não	Não	No (Não	Sim	Não
AWS Tax Settings	Sim	No (Não	Não	No (Não	Sim	Não
AWS Telco Network Builder	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Textract	Sim	No (Não	Não	No (Não	Sim	Não
Amazon Timestream	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon Timestream Influxdb	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS API do Tiros (para o Reachability Analyzer)	Sim	No (Não	Não	Não	Não	Não
Amazon Transcribe	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Transfer Family	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Transform	Sim	Yes (Sim	No (Não	No (Não	Sim	Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
Amazon Translate	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS Trusted Advisor	⚠ Parcial (Informações)	✓ Sim	✗ No (Não	✗ Não	⚠ Parcial	✓ Sim
AWS Notificações ao usuário da	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
AWS Contatos de notificações de usuários da	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✗ Não
AWS Assinaturas de usuário da	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ No (Não
Acesso Verificado pela AWS	✓ Sim	✗ No (Não	✗ Não	✗ No (Não	✓ Sim	✗ Não
Amazon Verified Permissions	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ Não
Amazon Virtual Private Cloud (Amazon VPC)	✓ Sim	⚠ Parcial (Informações)	⚠ Parcial (Informações)	✓ Sim	✓ Yes (Sim	⚠ Parcial (Informações)
Amazon VPC Lattice	✓ Sim	✓ Yes (Sim	✗ No (Não	✓ Yes (Sim	✓ Yes (Sim	✓ Sim
Serviços do Amazon VPC Lattice	✓ Sim	✓ Yes (Sim	✗ No (Não	✗ No (Não	✓ Sim	✗ No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS WAF	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS WAF Classic	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS WAF Regional	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
AWS Well-Architected Tool	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não
AWS Wickr	Yes (Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon WorkMail	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon WorkMail Message Flow	Sim	Yes (Sim	No (Não	No (Não	Sim	Não
Amazon WorkSpaces	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
AWS Instâncias gerenciadas pelo WorkSpaces	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Não
Amazon WorkSpaces Secure Browser	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	Sim
Amazon WorkSpaces Thin Client	Sim	Yes (Sim	No (Não	Yes (Sim	Yes (Sim	No (Não

Serviço	Ações do	Permissões em nível de recurso	Políticas baseadas em recursos	ABAC	Credenciais temporárias	Perfis vinculados ao serviço
AWS X-Ray	Sim	Parcial (Informações)	Não	Parcial (Informações)	Sim	Não

Mais informações

AWS CloudTrail

O CloudTrail é compatível com políticas baseadas em recursos em canais, painéis e armazenamentos de dados de eventos do CloudTrail Lake usados para integrações com fontes de eventos fora da AWS.

Amazon CloudWatch

Os perfis vinculados ao serviço do CloudWatch não podem ser criados usando o Console de gerenciamento da AWS e só oferecem suporte ao atributo Ações de alarme.

AWS CodeBuild

O CodeBuild é compatível com o compartilhamento de recursos entre contas usando o AWS RAM.

O CodeBuild é compatível com ABAC para ações baseadas em projetos.

AWS Config

O AWS Config é compatível com permissões em nível de recurso para agregação de dados de várias contas e várias regiões e regras do AWS Config. Para obter uma lista de recursos com suporte, consulte a seção Multi-Account Multi-Region Data Aggregation e a seção AWS Config Rules do Guia de APIs do AWS Config.

AWS Database Migration Service

É possível criar e modificar políticas que são anexadas às chaves de criptografia do AWS KMS criadas para criptografar dados migrados para endpoints de destino compatíveis. Os endpoints de destino compatíveis incluem o Amazon Redshift e o Amazon S3. Para obter mais informações, consulte Criar e usar chaves do AWS KMS para criptografar dados de destino do Amazon Redshift e Criar chaves do AWS KMS para criptografar objetos de destino do Amazon S3 no Guia do usuário do AWS Database Migration Service.

Amazon Elastic Compute Cloud

Os perfis vinculados ao serviço do Amazon EC2 só podem ser usados para os seguintes atributos: solicitações de instância spot, solicitações de frota spot, Amazon EC2 Fleets e inicialização rápida de instâncias do Windows.

Amazon Elastic Container Service

Somente algumas ações do Amazon ECS são compatíveis com permissões no nível do recurso.

AWS Elemental MediaPackage

O MediaPackage é compatível com perfis vinculados a serviço para publicar logs de acesso do cliente no CloudWatch, mas não para outras ações de API.

AWS Identity and Access Management

O IAM é compatível com apenas um tipo de política baseada em recurso, chamada política de confiança de um perfil, que é anexado a um perfil do IAM. Para obter mais informações, consulte Conceder permissões a um usuário para alternar perfis.

O IAM é compatível com controle de acesso baseado em etiquetas para a maioria dos recursos do IAM. Para obter mais informações, consulte Tags para recursos do AWS Identity and Access Management.

Somente algumas das ações de API do IAM podem ser chamadas com credenciais temporárias. Para obter mais informações, consulte Comparação de suas opções de API.

AWS IoT

Dispositivos conectados ao AWS IoT são autenticados usando certificados X.509 ou identidades do Amazon Cognito. É possível anexar políticas do AWS IoT a um certificado X.509 ou uma

identidade Amazon Cognito para controlar o que o dispositivo está autorizado a fazer. Para obter mais informações, consulte Segurança e identidade da AWS IoT no Guia do desenvolvedor da AWS IoT.

AWS Lambda

O Lambda é compatível com o controle de acesso por atributo (ABAC) para funções, mapeamentos da origem do evento e configurações de assinatura de código. Não há suporte a camadas. Para obter mais informações, consulte Usar controle de acesso baseado em atributos no Lambda.

O Lambda não tem perfis vinculados a serviços, mas o Lambda@Edge tem. Para obter mais informações, consulte Perfis vinculados ao serviço para o Lambda@Edge no Guia do desenvolvedor do Amazon CloudFront.

Amazon Lightsail

LightsailO é parcialmente compatível com permissões no nível do recurso e com ABAC. Para obter mais informações, consulte Ações, recursos e chaves de condição do Amazon Lightsail.

Amazon Managed Streaming for Apache Kafka (MSK)

É possível anexar uma política de cluster a um cluster do Amazon MSK que tenha sido configurado para conectividade com várias VPCs.

Aprovação Multilateral

Para fontes de identidade e equipes de aprovação multilateral, você pode usar as três chaves de condição ABAC: `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` e `aws:TagKeys`.

Para sessões, você somente pode usar a chave de condição `aws:ResourceTag/key-name` porque:

- Sessões não são recursos diretamente marcáveis.
- As sessões herdam as tags da equipe de aprovação a elas associada.

AWS Network Manager

AWSO Cloud WAN também é compatível com perfis vinculados a serviço. Para obter mais informações, consulte AWS Cloud WAN service-linked roles no Guia do AWS Cloud WAN para Amazon VPC.

Amazon Redshift Sem Servidor

O Redshift Serverless é compatível com políticas baseadas em recursos para compartilhar snapshots.

Amazon Relational Database Service

O Amazon Aurora é um mecanismo de banco de dados relacional gerenciado compatível com o MySQL e o PostgreSQL. É possível escolher o Aurora MySQL ou o Aurora PostgreSQL como opção de mecanismo de banco de dados durante a configuração de novos servidores de banco de dados por meio do Amazon RDS. Para obter mais informações, consulte Gerenciamento de identidade e acesso no Amazon Aurora no Guia do usuário do Amazon Aurora.

Amazon Rekognition

As políticas baseadas em recursos só são permitidas para copiar modelos de Amazon Rekognition Custom Labels.

Grupos de recursos da AWS

Os usuários podem assumir um perfil com uma política que permita operações de grupos de recurso.

Amazon SageMaker AI

Os perfis vinculados ao serviço estão atualmente disponíveis para tarefas de treinamento do SageMaker AI Studio e do SageMaker AI.

AWS Security Token Service

AWS STSO não tem “recursos”, mas permite a restrição de acesso de maneira semelhante aos usuários. Para obter mais informações, consulte Negação de acesso a credenciais de segurança temporárias por nome.

Somente algumas das operações da API para o AWS STS oferecem suporte a chamadas com credenciais temporárias. Para obter mais informações, consulte Comparação de suas opções de API.

Amazon Simple Email Service

Você só pode usar permissões no nível de recurso em instruções de política que se referem a ações relacionadas ao envio de e-mail, como `ses:SendEmail` ou `ses:SendRawEmail`. Para declarações de política que se referem a todas as outras ações, o elemento Resource só pode conter `*`.

Apenas a API do Amazon SES é compatível com credenciais de segurança temporárias. A interface SMTP do Amazon SES não é compatível com credenciais do SMTP derivadas de credenciais de segurança temporárias.

Amazon Simple Storage Service

O Amazon S3 é compatível com autorização baseada em tag para buckets de diretório, recursos de objeto, Lente de Armazenamento do Amazon S3, grupos da Lente de Armazenamento e pontos de acesso.

O Amazon S3 é compatível com perfis vinculados a serviço da Lente de Armazenamento do Amazon S3.

AWS Trusted Advisor

O acesso da API ao Trusted Advisor é feito por meio da API do Suporte e é controlado por políticas do IAM do Suporte.

Amazon Virtual Private Cloud

A Amazon VPC permite anexar uma única política de recursos a um endpoint da VPC para restringir o que pode ser acessado por meio desse endpoint. Para obter mais informações sobre o uso de políticas baseadas em recursos para controlar o acesso a recursos a partir de endpoints específicos da Amazon VPC, consulte Controlar o acesso a serviços usando políticas de endpoint no Guia do AWS PrivateLink.

AWS X-Ray

O X-Ray não é compatível com permissões no nível de recurso para todas as ações.

O X-Ray é compatível com o acesso baseado em etiquetas para grupos e regras de amostragem.

AWS Signature Version 4 para solicitações de API

Important

Caso use um SDK da AWS (consulte Sample Code and Libraries) ou a ferramenta da AWS Command Line Interface (AWS CLI) para enviar solicitações de API para a AWS, você pode pular o processo de assinatura, pois os clientes de SDK e CLI autenticarão suas solicitações

usando as chaves de acesso fornecidas por você. A menos que haja um bom motivo para não usar, é recomendável usar sempre um SDK ou a CLI.
Em regiões que oferecem suporte a várias versões de assinatura, assinar manualmente as solicitações significa que é necessário especificar qual versão de assinatura está sendo usada. Quando você fornece solicitações para pontos de acesso multirregionais, os SDKs e a CLI automaticamente alternam para o uso do Signature Version 4A, sem configuração adicional.

As informações de autenticação enviadas em uma solicitação devem incluir uma assinatura. O . AWS Signature Version 4 (SigV4) é o protocolo de assinatura da AWS para adicionar informações de autenticação às solicitações de API da AWS.

Você não usa sua chave de acesso secreta para assinar solicitações da API. Em vez disso, você usa o processo de assinatura da SigV4. A assinatura de solicitações envolve:

1. A criação de uma solicitação canônica com base nos detalhes da solicitação.
2. O cálculo uma assinatura usando suas credenciais da AWS.
3. Adicionar essa assinatura à solicitação como um cabeçalho de autorização.

AWSA , em seguida, replica esse processo e verifica a assinatura, concedendo ou negando o acesso adequadamente.

A Symmetric SigV4 exige que você derive uma chave que tenha como escopo um único serviço da AWS, em uma única região da AWS, em um determinado dia. Isso torna a chave e a assinatura calculada diferentes para cada região, o que significa que você precisa saber a região à qual a assinatura se destina.

A Assymmetric Signature Versão 4 (SigV4a) é uma extensão compatível com assinatura por um novo algoritmo e geração de assinaturas individuais que podem ser verificadas em mais de uma região da AWS. Com a SIGv4a, você pode assinar uma solicitação para várias regiões com roteamento e failover sem problemas entre regiões. Quando você usa o AWS SDK ou a AWS CLI para invocar uma funcionalidade que exige assinatura multirregional, o tipo de assinatura é alterado automaticamente para usar a Sigv4a. Para obter detalhes, consulte Como a AWS SigV4 funciona.

Como o SigV4 da AWS funciona

As seguintes etapas descrevem o processo geral de computação de uma assinatura com a SigV4:

1. A string para assinar depende do tipo de solicitação. Por exemplo, ao usar o cabeçalho HTTP Authorization ou os parâmetros de consulta para autenticação, utilize uma combinação de elementos de solicitação para criar a string a ser assinada. Para uma solicitação HTTP POST, a política POST na solicitação é a string que você assina.
2. A chave de assinatura é uma série de cálculos, com o resultado de cada etapa inserido na próxima. A etapa final é a chave de assinatura.
3. Ao receber uma solicitação autenticada, o serviço da AWS recria a assinatura usando as informações de autenticação contidas na solicitação. Se as assinaturas corresponderem, o serviço processará a solicitação. Caso contrário, rejeitará a solicitação.

Para obter mais informações, consulte Elementos de uma assinatura de solicitação de API da AWS.

Como a AWS SigV4 funciona

A Sigv4a usa assinaturas assimétricas baseadas em criptografia com chave pública-privada. A SigV4a passa por um processo de derivação de credenciais cujo escopo é semelhante ao da SigV4, exceto que a SigV4a usa a mesma chave para assinar todas as solicitações sem precisar derivar uma chave de assinatura distinta de acordo com a data, o serviço e a região. Um par de chaves de Algoritmo de Assinatura Digital de Curva Elíptica (ECDSA) pode ser derivado da chave de acesso secreta da AWS já existente.

O sistema usa criptografia assimétrica para verificar assinaturas multirregionais, de modo que a AWS só precise armazenar suas chaves públicas. As chaves públicas não são secretas e não podem ser usadas para assinar solicitações. Assinaturas assimétricas são necessárias para solicitações de API multirregionais, por exemplo, com os pontos de acesso multirregionais do Amazon S3.

As seguintes etapas descrevem o processo geral de computação de uma assinatura com a SigV4a:

1. A string para assinar depende do tipo de solicitação. Por exemplo, ao usar o cabeçalho HTTP Authorization ou os parâmetros de consulta para autenticação, utilize uma combinação de elementos de solicitação para criar a string a ser assinada. Para uma solicitação HTTP POST, a política POST na solicitação é a string que você assina.
2. A chave de assinatura é derivada de uma chave de acesso secreta da AWS por uma série de cálculos, sendo que o resultado de cada etapa é inserido na próxima etapa. A etapa final produz o par de chaves.
3. Quando um serviço da AWS recebe uma solicitação assinada com a SigV4a, a AWS verifica a assinatura usando somente a metade pública do par de chaves. Se a assinatura for válida, a

solicitação será autenticada e o serviço processará a solicitação. Solicitações com assinaturas inválidas são rejeitadas.

Para obter mais informações sobre a SigV4a para solicitações de API multirregionais, consulte o projeto sigv4a-signing-examples no GitHub.

Quando assinar solicitações

Ao escrever um código personalizado que envia solicitações de API para a AWS, é necessário incluir um código que assine as solicitações. Você poderá escrever um código personalizado porque:

- Você está trabalhando com uma linguagem de programação para a qual não há nenhum SDK da AWS.
- Você precisa ter controle total sobre como as solicitações são enviadas à AWS.

Enquanto as solicitações de API autenticam o acesso com o SigV4 da AWS, os SDKs da AWS e a AWS CLI autenticam suas solicitações usando as chaves de acesso que você fornece. Para obter mais informações sobre a autenticação com SDKs da AWS e a AWS CLI, consulte Recursos adicionais.

Por que as solicitações são assinadas

O processo de assinatura ajuda a proteger as solicitações das seguintes formas:

- Verificar a identidade do solicitante

  As solicitações autenticadas exigem uma assinatura que você cria usando as chaves de acesso (ID da chave de acesso, chave de acesso secreta). Se você usar credenciais de segurança temporárias, os cálculos da assinatura também exigirão um token de segurança. Para obter mais informações, consulte Credenciais de segurança da AWS, Acesso programático.
- Proteger dados em trânsito

  Para evitar violação de uma solicitação enquanto ela estiver em trânsito, alguns dos elementos de solicitação são usados para calcular um hash (resumo) da solicitação, e o valor de hash resultante é incluído como parte da solicitação. Ao receber a solicitação, um AWS service (Serviço da AWS) usa as mesmas informações para calcular um hash e o compara com o valor de hash na solicitação. Se os valores não coincidirem, a AWS nega a solicitação.
- Proteger contra ataques potenciais de replay

Na maioria dos casos, uma solicitação deve chegar à AWS no lapso de cinco minutos a partir da marca de tempo na solicitação. Caso contrário, a AWS negará a solicitação.

AWSO SigV4 da pode ser expresso no cabeçalho HTTP Authorization ou como uma string de consulta no URL. Para obter mais informações, consulte Métodos de autenticação.

Recursos adicionais

- Para obter mais informações sobre o processo de assinatura do SigV4 para diferentes serviços, consulte Exemplos de assinatura de solicitação.
- Para configurar credenciais para acesso programático à AWS CLI, consulte Authentication and access credentials no AWS Command Line Interface User Guide.
- Os AWS SDKs incluem código-fonte no GitHub para assinar solicitações de API da AWS. Para obter exemplos de código, consulte Projetos de exemplo em repositório de amostras da AWS.
  - AWS SDK para .NET – AWSSigV4Signer.cs
  - AWS SDK para C++ – AWSAuthV4Signer.cpp
  - AWS SDK para Go – sigv4.go
  - AWS SDK para Java – BaseAws4Signer.java
  - AWS SDK para JavaScript – signature-v4
  - AWS SDK para PHP – SignatureV4.php
  - AWS SDK para Python (Boto) – signers.py
  - AWS SDK para Ruby – signer.rb
- Se você encontrar erros ao assinar solicitações, consulte Solucionar problemas de assinatura do Signature Version 4 para solicitações de API da AWS.

Elementos de uma assinatura de solicitação de API da AWS

> **Important**
>
> A menos que você esteja usando AWS SDKs ou a CLI, é necessário escrever código para calcular assinaturas que forneçam informações de autenticação em suas solicitações. O cálculo da assinatura no AWS Signature Versão 4 pode ser uma tarefa complexa, então é recomendável usar os AWS SDKs ou a CLI sempre que possível.

Cada solicitação HTTP/HTTPS que usa a assinatura do Signature Version 4 deve conter esses elementos.

Elementos

- Especificação de endpoint
- Ação
- Parâmetros de ação
- Data
- Informações de autenticação

Especificação de endpoint

Especifica o nome DNS do endpoint ao qual você envia a solicitação. Esse nome geralmente contém o código do serviço e a região. Por exemplo, o nome do endpoint para o Amazon DynamoDB na região `us-east-1` é `dynamodb.us-east-1.amazonaws.com`.

Para solicitações de HTTP/1.1, é necessário incluir o cabeçalho `Host`. Para solicitações HTTP/2, você pode incluir o cabeçalho `:authority` ou o cabeçalho `Host`. Use apenas o cabeçalho `:authority` em conformidade com a especificação HTTP/2. Nem todos os serviços oferecem suporte a solicitações HTTP/2.

Para obter os endpoints compatíveis com cada serviço, consulte Endpoints e cotas de serviço na Referência geral da AWS.

Ação

Especifica uma ação de API para o serviço. Por exemplo, a ação `CreateTable` do DynamoDB ou a ação `DescribeInstances` do Amazon EC2.

Para ver as ações compatíveis com cada serviço, consulte a Referência de autorização do serviço.

Parâmetros de ação

Especifica os parâmetros da ação especificada na solicitação. Cada ação de API da AWS tem um conjunto de parâmetros obrigatórios e opcionais. A versão da API geralmente é um parâmetro obrigatório.

Para ver os parâmetros compatíveis com uma ação de API, consulte a Referência de API do serviço.

Data

Especifica a data e a hora da solicitação. Incluir a data e a hora na solicitação ajuda a evitar que terceiros interceptem sua solicitação e a enviem novamente mais tarde. A data especificada no escopo de credenciais deve corresponder à data da solicitação.

O carimbo de data e hora deve estar em UTC e usar o seguinte formato: ISO 8601 AAAAMMDDTHHMMSSZ. Por exemplo, `20220830T123600Z`. Não inclua milissegundos na marca de tempo.

Você pode usar um cabeçalho `date` ou `x-amz-date` ou incluir `x-amz-date` como um parâmetro de consulta. Se não conseguirmos encontrar um cabeçalho `x-amz-date`, procuraremos um cabeçalho `date`.

Informações de autenticação

Cada solicitação enviada deve incluir as informações a seguir. A AWS usa essas informações para garantir a validade e a autenticidade da solicitação.

- Algoritmo: o algoritmo que você está usando como parte do processo de assinatura.
  - SigV4: use `AWS4-HMAC-SHA256` para especificar o Signature Version 4 com o algoritmo de hash `HMAC-SHA256`.
  - SigV4a: use `AWS4-ECDSA-P256-SHA256` para especificar o algoritmo de hash `ECDSA-P256-SHA-256`.
- Credencial: uma string que é formada concatenando o ID da chave de acesso e os componentes do escopo de credencial.
  - SigV4: o escopo da credencial inclui o ID da chave de acesso, a data no formato AAAAMMDD, o código da região, o código do serviço e a string `aws4_request` de término separados por barras (/). O código da região, o código do serviço e a string de término devem usar caracteres minúsculos.

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
    ```

  - SigV4a: o escopo da credencial inclui a data no formato AAAAMMDD, o nome do serviço e a string `aws4_request` de término separados por barras (/). Observe que o escopo da credencial não inclui a região, pois ela é englobada em um cabeçalho `X-Amz-Region-Set` separado.

    ```
    AKIAIOSFODNN7EXAMPLE/YYYYMMDD/service/aws4_request
    ```

- Cabeçalhos assinados: os cabeçalhos HTTP a serem incluídos na assinatura, separados por ponto e vírgula (;). Por exemplo, `host;x-amz-date`.

  Para SigV4a, você deve incluir um cabeçalho de conjunto de regiões que especifique o conjunto de regiões em que a solicitação será válida. O cabeçalho `X-Amz-Region-Set` é especificado como uma lista de valores separados por vírgula. O exemplo a seguir mostra um cabeçalho de região que permite fazer uma solicitação nas regiões us-east-1 e us-west-1.

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  Você pode usar curingas (*) em regiões para especificar várias regiões. No exemplo a seguir, o cabeçalho permite que uma solicitação seja feita tanto na região us-west-1 quanto na us-west-2.

  ```
  X-Amz-Region-Set=us-west-*
  ```

- Assinatura: uma string codificada em hexadecimal que representa a assinatura calculada. Você deve calcular a assinatura com o algoritmo especificado no parâmetro `Algorithm`.

Para obter mais informações, consulte Métodos de autenticação

Métodos de autenticação

> **Important**
>
> A menos que você esteja usando AWS SDKs ou a CLI, é necessário escrever código para calcular assinaturas que forneçam informações de autenticação em suas solicitações. O cálculo da assinatura no AWS Signature Versão 4 pode ser uma tarefa complexa, então é recomendável usar os AWS SDKs ou a CLI sempre que possível.

É possível expressar informações de autenticação usando um dos métodos a seguir.

Cabeçalho HTTP de autorização

O cabeçalho HTTP `Authorization` é o método mais usado para autenticar uma solicitação. Todas as operações da API REST (exceto para carregamentos baseados em navegador que usam solicitações `POST`) requerem esse cabeçalho.

Os exemplos a seguir mostram o valor do cabeçalho `Authorization` para SigV4 e SigV4a. As quebras de linha foram adicionadas a este exemplo somente para facilitar a leitura. Em seu código, o cabeçalho deve ser uma string contínua. Não use vírgula entre o algoritmo e a credencial, mas os outros elementos devem ser separados por vírgulas.

Example SigV4

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Example Sigv4a

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/s3/aws4_request,
SignedHeaders=host;range;x-amz-date;x-amz-region-set,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

A tabela a seguir descreve os vários componentes do valor do cabeçalho de autorização do exemplo anterior:

Componente	Descrição
Autorização	O algoritmo que foi usado para calcular a assinatura. • SigV4: use `AWS4-HMAC-SHA256` . Esta string identifica o AWS SigV4 (`AWS4`) e o algoritmo `HMAC-SHA256` . • SigV4a: use `AWS4-ECDSA-P256-SHA256` . Esta string identifica o AWS SigV4 (`AWS4`) e o algoritmo `ECDSA-P256-SHA-256` .
Credential	Seu ID de chave de acesso e as informações do escopo.

Componente	Descrição

	• SigV4: inclua a data, a região e o serviço que foram usados para calcular a assinatura. Essa string tem a seguinte forma:
	
	`<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request`
	• SigV4a: inclua a data e o serviço que foram usados para calcular a assinatura. Essa string tem a seguinte forma:
	
	`<your-access-key-id>/<date>/ <aws-service>/aws4_request`
	
	O valor de **<date>** é especificado usando o formato AAAAMMDD. O valor de **<aws-service>** é S3 ao enviar uma solicitação ao Amazon S3.

SignedHeaders	Uma lista separada por ponto e vírgula dos cabeçalhos de solicitação usados para calcular a assinatura. A lista contém somente nomes de cabeçalho, e os nomes dos cabeçalhos devem estar em letras minúsculas. Por exemplo, .: `host;range;x-amz-date` Para SigV4a, você deve incluir um cabeçalho de conjunto de regiões que especifique o conjunto de regiões em que a solicitação será válida. O cabeçalho X-Amz-Region-Set é especificado como uma lista de valores separados por vírgula.

Componente	Descrição
Signature	A assinatura de 256 bits expressa como 64 caracteres hexadecimais minúsculos. Por exemplo, .:`fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024` Os cálculos da assinatura variam de acordo com a opção escolhida para transferência de carga útil.

Parâmetros de string de consulta

É possível usar uma string de consulta para expressar uma solicitação completa em um só URL. Nesse caso, utilize parâmetros de consulta para fornecer informações de solicitação, inclusive as informações de autenticação. Como a solicitação de assinatura faz parte do URL, esse tipo de URL muitas vezes é chamado de URL pré-assinado. É possível usar URLs pré-assinados para incorporar links clicáveis em HTML, que podem ser válidos por até sete dias. Para obter mais informações, consulte Authenticating Requests: Using Query Parameters (AWS Signature Version 4) na Referência de APIs do Amazon S3.

Os exemplos a seguir mostram URLs pré-assinados para SigV4 e SigV4a. As quebras de linha foram adicionadas a este exemplo somente para facilitar a leitura:

Example SigV4

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/<region>/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

Example Sigv4a

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/test.txt ?
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256 &
X-Amz-Credential=<your-access-key-id>/20240721/s3/aws4_request &
```

```
X-amz-Region-Set=<regionset> &
X-Amz-Date=20240721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host;x-amz-region-set &
X-Amz-Signature=<signature-value>
```

> **Note**
>
> O valor de `X-Amz-Credential` no URL exibe o caractere "/" somente para facilitar a leitura. Na prática, deve ser codificado como %2F. Por exemplo: `&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request`

A tabela a seguir descreve os parâmetros de consulta no URL que fornecem informações de autenticação.

Nome de parâmetro de string de consulta	Descrição
X-Amz-Algorithm	A versão da assinatura da AWS e o algoritmo que você usou para calcular a assinatura. • SigV4: use `AWS4-HMAC-SHA256` . Esta string identifica o AWS SigV4 (`AWS4`) e o algoritmo `HMAC-SHA256` . • SigV4a: use `AWS4-ECDSA-P256-SHA256` . Esta string identifica o AWS SigV4 (`AWS4`) e o algoritmo `ECDSA-P256-SHA-256` .
X-Amz-Credential	Além do ID da chave de acesso, esse parâmetro também fornece o escopo para o qual a assinatura é válida. O valor deve corresponder ao escopo usado nos cálculos de assinatura, abordados na seção a seguir. • SigV4: a forma geral para este valor de parâmetro é: `<your-access-key-id>/`

Nome de parâmetro de string de consulta	Descrição
	`<date>/<AWS Region>/<AWS-service>/aws4_request` Por exemplo, .: `AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request` • SigV4a: a forma geral para este valor de parâmetro é: `<your-access-key-id>/<date>/<AWS-service>/aws4_request` Por exemplo, .: `AKIAIOSFODNN7EXAMPLE/20130721/s3/aws4_request` A região para SigV4a é definida no cabeçalho do conjunto de regiões `X-Amz-Region-Set` . Para obter uma lista de strings regionais da AWS, consulte Regional Endpoints na Referência geral da AWS.
X-Amz-Region-Set	O conjunto de regiões em que a solicitação será válida. O cabeçalho x-amz-region-set é especificado como uma lista de valores separados por vírgula.
X-Amz-Date	O formato de data e hora deve seguir o padrão ISO 8601 e deve ter a formatação `yyyyMMddTHHmmssZ` . Por exemplo, se a data e a hora forem “08/01/2016 15:32:41.982-700”, elas primeiro deverão ser convertidas em UTC (Tempo Universal Coordenado) e depois enviadas como “20160801T223241Z”.

Nome de parâmetro de string de consulta	Descrição
X-Amz-Expires	Fornece o período, em segundos, de validade do URL pré-assinado gerado. Por exemplo, 86400 (24 horas). Esse valor é um inteiro. O valor mínimo que você pode definir é 1 e o máximo é 604800 (sete dias). O URL pré-assinado pode ser válido por no máximo sete dias, pois a chave de assinatura usada no cálculo da assinatura é válida por até sete dias.
X-Amz-SignedHeaders	Lista os cabeçalhos usados para calcular a assinatura. Os seguintes cabeçalhos são obrigatórios para os cálculos da assinatura: • O cabeçalho do host HTTP. • Todo cabeçalho x-amz-* que você pretende adicionar à solicitação. • Para SigV4a, X-Amz-Region-Set é necessário para especificar as regiões em que a solicitação pode ser feita. Para maior segurança, é necessário assinar todos os cabeçalhos de solicitação que pretende incluir na solicitação.
X-Amz-Signature	Fornece a assinatura para autenticar a solicitação. Essa assinatura deve corresponder à assinatura calculada pelo serviço; caso contrário, o serviço negará a solicitação. Por exemplo, ., `733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7` Os cálculos de assinatura serão descritos na seção a seguir.

Nome de parâmetro de string de consulta	Descrição
X-Amz-Security-Token	Parâmetro de credencial opcional ao usar credenciais provenientes do serviço STS.

Crie uma solicitação assinada de API da AWS

> **Important**
>
> Caso use um SDK da AWS (consulte Código de exemplo e bibliotecas) ou a ferramenta AWS Command Line Interface (AWS CLI) para enviar solicitações de API à AWS, você pode pular esta seção, pois os clientes do SDK e CLI autenticam as solicitações usando as chaves de acesso fornecidas por você. A menos que haja um bom motivo para não usar, é recomendável usar sempre um SDK ou a CLI.
> Em regiões compatíveis com várias versões de assinatura, assinar as solicitações manualmente significa que é necessário especificar qual versão de assinatura está sendo usada. Quando você fornece solicitações para pontos de acesso multirregionais, os SDKs e a CLI automaticamente alternam para o uso do Signature Version 4A, sem configuração adicional.

Você pode usar o protocolo de assinatura SigV4 da AWS para criar uma solicitação assinada para solicitações de API da AWS.

1. A criação de uma solicitação canônica com base nos detalhes da solicitação.
2. O cálculo uma assinatura usando suas credenciais da AWS.
3. Adicionar essa assinatura à solicitação como um cabeçalho de autorização.

AWSA , em seguida, replica esse processo e verifica a assinatura, concedendo ou negando o acesso adequadamente.

Para ver como você pode usar o AWS SigV4 para assinar solicitações de API, consulte Exemplos de assinatura de solicitação.

A tabela a seguir descreve as funções usadas no processo de criação de uma solicitação assinada. É necessário implementar o código para essas funções. Para obter mais informações, consulte os exemplos de código nos AWS SDKs.

Função	Descrição
`Lowercase()`	Converta a string em letras minúsculas.
`Hex()`	Codificação de base 16 em letras minúsculas.
`SHA256Hash()`	Função de hash criptográfico do Secure Hash Algorithm (SHA).
`HMAC-SHA256()`	Calcula o HMAC usando o algoritmo SHA256 com a chave de assinatura fornecida. Esta é a assinatura final quando você assina com o SigV4.
`ECDSA-Sign`	Assinatura do Algoritmo de assinatura digital de curva elíptica (ECDSA) calculada usando assinaturas assimétricas baseadas em criptografia de chave pública-privada.
`KDF(K, Label, Context, L)`	Um KDF NIST SP800-108 no Modo Contador usando a função PRF HMAC-SHA256, conforme definido no NIST SP 800-108r1.
`Oct2Int(byte[ ])`	Uma função de octeto para inteiro, conforme descrito no ANSI X9.62.
`Trim()`	Remova qualquer espaço em branco inicial e final.
`UriEncode()`	O URI codifica cada byte. O UriEncode() deve aplicar as seguintes regras: • O URI codifica cada byte, exceto os caracteres não reservados: “A”-“Z”, “a”-“z”, “0”-“9”, “-”, “.”, “_” e “~”. • O caractere de espaço é um caractere reservado e deve ser codificado como “%20” (e não como “+”).

Função	Descrição
	• Cada byte codificado por URI é formado por um “%” e o valor hexadecimal de dois dígitos do byte. • As letras no valor hexadecimal devem estar em maiúsculas; por exemplo, “%1A”. • Codifique o caractere de barra, “/”, em todos os lugares, exceto no nome da chave do objeto. Por exemplo, se o nome da chave do objeto for `photos/Jan/sample.jpg` , a barra no nome da chave não está codificada. **Important** As funções UriEncode padrão fornecidas pela sua plataforma de desenvolvimento podem não funcionar devido às diferenças na implementação e à ambiguidade relacionada nos RFCs subjacentes. É recomendável escrever sua própria função UriEncode personalizada para certificar-se de que a codificação funcione. Para ver um exemplo de uma função UriEncode em Java, consulte Java Utilities no site do GitHub.

Note

Ao assinar as solicitações, você pode usar o AWS SigV4 ou o AWS SigV4a. A principal diferença entre os dois está na forma como a assinatura é calculada. Com o SigV4a, o

conjunto de regiões é incluído na string a ser assinada, mas não faz parte da etapa de derivação da credencial.

Solicitação de assinatura com credenciais de segurança temporárias

Em vez de usar credenciais de longo prazo para assinar uma solicitação, é possível usar credenciais de segurança temporárias fornecidas pelo AWS Security Token Service (AWS STS).

Ao usar credenciais de segurança temporárias, é necessário adicionar o código `X-Amz-Security-Token` ao cabeçalho de autorização ou incluí-lo na string de consulta para manter o token da sessão. Alguns serviços exigem que você adicione `X-Amz-Security-Token` à solicitação canônica. Outros serviços exigem apenas que você adicione `X-Amz-Security-Token` no final, depois de calcular a assinatura. Verifique a documentação de cada AWS service (Serviço da AWS) para obter requisitos específicos.

Resumo das etapas de assinatura

Criar uma solicitação canônica

Organize o conteúdo da solicitação (host, ação, cabeçalhos etc.) em um formato padrão canônico. A solicitação canônica é um recurso usado para criar a string para assinar. Para obter detalhes sobre a criação da solicitação canônica, consulte Elementos de uma assinatura de solicitação de API da AWS.

Criar um hash para a solicitação canônica

Faça o hash da solicitação canônica usando o mesmo algoritmo que você usou para criar o hash da carga útil. O hash da solicitação canônica é uma string de caracteres hexadecimais em minúsculas.

Criar uma string para assinar

Crie uma string para assinar com a solicitação canônica e informações adicionais, como o algoritmo, data de solicitação, escopo de credencial e o hash da solicitação canônica.

Derivar uma chave de assinatura

Use a chave de acesso secreta para derivar a chave usada para assinar a solicitação.

Calcular a assinatura

Realize uma operação de hash com chave na string para assinar usando a chave de assinatura derivada como a chave de hash.

Adicionar a assinatura à solicitação

Adicione a assinatura calculada a um cabeçalho de HTTP ou à string de consulta da solicitação.

Criar uma solicitação canônica

Para criar uma solicitação canônica, concatene as strings a seguir separadas por caracteres de linha nova. Isso ajuda a certificar-se de que a assinatura que você calcula possa corresponder à assinatura que a AWS calcula.

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod*: o método HTTP, como `GET`, `PUT`, `HEAD` e `DELETE`.
- *CanonicalUri*: a versão codificada em URI do componente de caminho absoluto do URI, começando com o código `/` que segue o nome do domínio e indo até o final da string ou até o caractere de ponto de interrogação (`?`) se você tiver parâmetros de string de consulta. Se o caminho absoluto estiver vazio, use um caractere de barra inclinada (`/`). O URI no exemplo a seguir, `/amzn-s3-demo-bucket/myphoto.jpg`, é o caminho absoluto, e você não codifica o `/` no caminho absoluto:

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket/myphoto.jpg
  ```

- *CanonicalQueryString*: os parâmetros da string de consulta codificados por URI. Codifique por URI cada nome e valor individualmente. Também é necessário classificar os parâmetros na string de consulta canônica em ordem alfabética pelo nome da chave. A classificação ocorre após a codificação. A string de consulta no seguinte exemplo de URI é:

  ```
  http://s3.amazonaws.com/amzn-s3-demo-bucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  A string de consulta canônica é como este exemplo (quebras de linha foram adicionadas ao exemplo para facilitar a leitura):

```
UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
UriEncode("max-keys")+"="+UriEncode("20") + "&" +
UriEncode("prefix")+"="+UriEncode("somePrefix")
```

Quando uma solicitação se destina a um sub-recurso, o valor do parâmetro de consulta correspondente é uma string vazia (""). Por exemplo, o URI a seguir identifica o sub-recurso `ACL` no bucket `amzn-s3-demo-bucket`:

```
http://s3.amazonaws.com/amzn-s3-demo-bucket?acl
```

Nesse caso, a CanonicalQueryString seria:

```
UriEncode("acl") + "=" + ""
```

Caso o URI não contenha um ?, não há strings de consulta na solicitação, e você define a string de consulta canônica como uma string vazia (""). Você ainda precisará incluir o caractere de linha nova ("\n").

- *`CanonicalHeaders`*: uma lista de cabeçalhos de solicitação com os respectivos valores. Os pares individuais de nome e valor do cabeçalho são separados pelo caractere de linha nova ("\n"). Veja a seguir um exemplo de um CanonicalHeader:

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

A lista CanonicalHeaders deve conter:

- Cabeçalho HTTP `host`
- Se o cabeçalho `Content-Type` estiver presente na solicitação, você deverá adicioná-lo à lista *`CanonicalHeaders`*.
- Também deverão ser adicionados todos os cabeçalhos `x-amz-*` que você pretende incluir na solicitação. Por exemplo, se você estiver usando credenciais de segurança temporárias, precisará incluir o `x-amz-security-token` na solicitação. É necessário adicionar esse cabeçalho à lista de *`CanonicalHeaders`*.

- Para SigV4a, você deve incluir um cabeçalho de conjunto de regiões que especifique o conjunto de regiões em que a solicitação será válida. O cabeçalho `X-Amz-Region-Set` é especificado como uma lista de valores separados por vírgula. O exemplo a seguir mostra um cabeçalho de região que permite fazer uma solicitação nas regiões us-east-1 e us-west-1.

  ```
  X-Amz-Region-Set=us-east-1,us-west-1
  ```

  Você pode usar curingas (*) em regiões para especificar várias regiões. No exemplo a seguir, o cabeçalho permite que uma solicitação seja feita tanto na região us-west-1 quanto na us-west-2.

  ```
  X-Amz-Region-Set=us-west-*
  ```

> **Note**
>
> O cabeçalho `x-amz-content-sha256` é obrigatório para solicitações da AWS do Amazon S3. Ele fornece um hash da carga da solicitação. Se não houver carga útil, será necessário fornecer o hash de uma string vazia.

Todo nome de cabeçalho deve:

- usar caracteres minúsculos.
- ser exibido em ordem alfabética.
- ser seguido por dois pontos (`:`).

Para valores, é necessário:

- remover todos os espaços à esquerda ou à direita.
- converter espaços sequenciais em um espaço único.
- separar os valores de um cabeçalho de múltiplos valores usando vírgulas.
- É necessário incluir na assinatura o cabeçalho host (HTTP/1.1) ou o cabeçalho :authority (HTTP/2) e quaisquer cabeçalhos `x-amz-*`. Opcionalmente, você pode incluir outros cabeçalhos padrão na assinatura, como content-type.

As funções `Lowercase()` e `Trim()` usadas neste exemplo estão descritas na seção anterior.

Veja a seguir um exemplo de string `CanonicalHeaders`. O nomes dos cabeçalhos estão em letras minúsculas e são classificados.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Para fins de cálculo de uma assinatura de autorização, somente o host e cabeçalhos `x-amz-*` são obrigatórios; no entanto, para evitar a adulteração de dados, considere incluir os cabeçalhos adicionais no cálculo da assinatura.
> Não inclua cabeçalhos hop-by-hop que são alterados com frequência durante o trânsito em um sistema complexo. Isso inclui todos os cabeçalhos de transporte voláteis que são alterados por proxies, balanceadores de carga e nós em um sistema distribuído, incluindo `connection`, `x-amzn-trace-id`, `user-agent`, `keep-alive`, `transfer-encoding`, `TE`, `trailer`, `upgrade`, `proxy-authorization` e `proxy-authenticate`.

- *`SignedHeaders`*: uma lista ordenada alfabeticamente, separada por ponto e vírgula, de nomes de cabeçalhos de solicitação em letras minúsculas. Os cabeçalhos de solicitação da lista são os mesmos cabeçalhos que você incluiu na string `CanonicalHeaders`. Para o exemplo anterior, o valor de *`SignedHeaders`* seria:

  ```
  host;x-amz-content-sha256;x-amz-date
  ```

- *`HashedPayload`*: uma string criada usando a carga útil no corpo da solicitação HTTP como entrada para uma função de hash. Esta string usa caracteres hexadecimais minúsculos.

  ```
  Hex(SHA256Hash(<payload>))
  ```

  Se não houver carga útil na solicitação, calcule um hash da string vazia. Por exemplo, ao recuperar um objeto usando uma solicitação `GET`, não haverá nada na carga útil.

  ```
  Hex(SHA256Hash(""))
  ```

> **Note**
>
> Para o Amazon S3, inclua a string literal `UNSIGNED-PAYLOAD` ao construir uma solicitação canônica e defina o mesmo valor como o cabeçalho `x-amz-content-sha256` ao enviar a solicitação.
> `Hex(SHA256Hash("UNSIGNED-PAYLOAD"))`

Criar um hash para a solicitação canônica

Crie um hash (resumo) da solicitação canônica usando o mesmo algoritmo que você usou para criar o hash da carga útil. O hash da solicitação canônica é uma string de caracteres hexadecimais em minúsculas.

Criar uma string para assinar

Para criar uma string para assinar, concatene as strings a seguir separadas por caracteres de linha nova. Não termine esta string com um caractere de linha nova.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *`Algorithm`*: o algoritmo usado para criar o hash da solicitação canônica.
  - SigV4: use `AWS4-HMAC-SHA256` para especificar o algoritmo de hash `HMAC-SHA256`.
  - SigV4a: use `AWS4-ECDSA-P256-SHA256` para especificar o algoritmo de hash `ECDSA-P256-SHA-256`.
- *`RequestDateTime`*: a data e a hora usadas no escopo da credencial. Esse valor é a hora UTC atual no formato ISO 8601 (por exemplo, `20130524T000000Z`).
- *`CredentialScope`*: o escopo de credencial, que restringe a assinatura resultante à região e ao serviço especificados.
  - SigV4: as credenciais incluem o ID da chave de acesso, a data no formato `YYYYMMDD`, o código da região, o código do serviço e a string `aws4_request` de término separados por barras (/). O código da região, o código do serviço e a string de término devem usar caracteres minúsculos. A string tem o seguinte formato: `YYYYMMDD/region/service/aws4_request`.

  - SigV4a: as credenciais incluem a data no formato `YYYYMMDD`, o nome do serviço e a string `aws4_request` de término separados por barras (/). Observe que o escopo da credencial não inclui a região, pois ela é englobada em um cabeçalho `X-Amz-Region-Set` separado. A string tem o seguinte formato: `YYYYMMDD/service/aws4_request`.
- *`HashedCanonicalRequest`*: o hash da solicitação canônica, calculada na etapa anterior.

Veja a seguir um exemplo de string para assinar.

```
"<Algorithm>" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(<Algorithm>(<CanonicalRequest>))
```

Derivar uma chave de assinatura

Para derivar uma chave de assinatura, escolha um dos processos a seguir para calcular uma chave de assinatura para SigV4 ou SigV4a.

Derivar uma chave de assinatura para SigV4

Para derivar uma chave de assinatura para SigV4, realize uma sequência de operações de hash com chave (HMAC) no serviço, região e data da solicitação usando sua chave de acesso secreta da AWS como a chave para a operação inicial de hashing.

Para cada etapa, chame a função de hash com a chave e os dados necessários. O resultado de cada chamada para a função de hash torna-se a entrada para a próxima chamada para a função de hash.

O exemplo a seguir mostra como você deriva a `SigningKey` usada na próxima seção desse procedimento, mostrando a ordem na qual sua entrada é concatenada e transformada em hash. O código `HMAC-SHA256` é a função de hash usada para fazer o hash dos dados, conforme mostrado.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Entrada obrigatória

- `Key`: uma string que contém sua chave de acesso secreta.

- `Date`: uma string que contém a data usada no escopo da credencial, no formato AAAAMMDD.
- `Region`: uma string que contém o código da região (por exemplo, `us-east-1`).

  Para obter uma lista de regiões de strings, consulte Regional endpoints, na Referência geral da AWS.
- `Service`: uma string que contém o código do serviço (por exemplo, `ec2`).
- A string para assinar que você criou na etapa anterior.

Derivar uma chave de assinatura para SigV4

1. Concatene `"AWS4"` e a chave de acesso secreta. Chame a função de hash com a string concatenada como chave e a string de data como os dados.

   ```
   DateKey = hash("AWS4" + Key, Date)
   ```
2. Chame a função de hash com o resultado da chamada anterior como chave e a string de região como os dados.

   ```
   DateRegionKey = hash(kDate, Region)
   ```
3. Chame a função de hash com o resultado da chamada anterior como a chave e a string de serviço como os dados.

   O código do serviço é definido pelo serviço. Você pode usar get-products na AWS Pricing CLI para retornar o código de serviço de um serviço.

   ```
   DateRegionServiceKey = hash(kRegion, Service)
   ```
4. Chame a função de hash com o resultado da chamada anterior como a chave e “aws4_request” como os dados.

   ```
   SigningKey = hash(kService, "aws4_request")
   ```

Derivar uma chave de assinatura para SigV4a

Para criar uma chave de assinatura para SigV4a, use o processo a seguir para derivar um par de chaves da chave de acesso secreta. Para obter um exemplo de implementação dessa derivação, consulte a implementação da biblioteca C99 da autenticação do lado do cliente da AWS

```
n = [NIST P-256 elliptic curve group order]
G = [NIST P-256 elliptic curve base point]
label = "AWS4-ECDSA-P256-SHA256"

akid = [AWS access key ID as a UTF8 string]
sk = [AWS secret access Key as a UTF8 Base64 string]

input_key = "AWS4A" || sk
count = 1
while (counter != 255) {
  context = akid || counter // note: counter is one byte
  key = KDF(input_key, label, context, 256)
  c = Oct2Int(key)
  if (c > n - 2) {
    counter++
  } else {
    k = c + 1   // private key
    Q = k * G   // public key
  }
}

if (c < 255) {
  return [k, Q]
} else {
  return FAILURE
}
```

Calcular a assinatura

Depois de derivar a chave de assinatura, calcule a assinatura para adicionar à sua solicitação. Este procedimento varia de acordo com a versão de assinatura que você usa.

Para calcular uma assinatura para SigV4

1. Chame a função de hash com o resultado da chamada anterior como a chave e a string para assinar como os dados. Use a chave de assinatura derivada como a chave de hash para esta operação. O resultado é a assinatura como valor binário.

   ```
   signature = hash(SigningKey, string-to-sign)
   ```

2. Converta a assinatura da representação binária em hexadecimal, em caracteres minúsculos.

Para calcular uma assinatura para SigV4a

1. Usando o algoritmo de assinatura digital (ECDSA P-256), assine a string para assinar que você criou na etapa anterior. A chave usada para essa assinatura é a chave assimétrica privada derivada da chave de acesso secreta, conforme descrito acima.

```
signature = base16(ECDSA-Sign(k, string-to-sign))
```

2. Converta a assinatura da representação binária em hexadecimal, em caracteres minúsculos.

Adicionar a assinatura à solicitação

Adicione a assinatura calculada à sua solicitação.

Example Exemplo: cabeçalho de autorização

SigV4

O exemplo a seguir mostra um cabeçalho `Authorization` para a ação `DescribeInstances` usando o AWS SigV4. Para facilitar a leitura, este exemplo está formatado com quebras de linha. Em seu código, deve ser uma string contínua. Não há vírgula entre o algoritmo e `Credential`. Porém, os outros elementos devem ser separados por vírgulas.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

Sigv4a

O exemplo a seguir mostra um cabeçalho Authorization para a ação `CreateBucket` usando o AWS SigV4a. Para facilitar a leitura, este exemplo está formatado com quebras de linha. Em seu código, deve ser uma string contínua. Não há vírgula entre o algoritmo e a credencial. Porém, os outros elementos devem ser separados por vírgulas.

```
Authorization: AWS4-ECDSA-P256-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request,
SignedHeaders=host;x-amz-date;x-amz-region-set,
Signature=calculated-signature
```

Example Exemplo: solicitação com parâmetros de autenticação na string de consulta

SigV4

O exemplo a seguir mostra uma consulta para a ação `DescribeInstances` usando o AWS SigV4 que contém as informações de autenticação. Para facilitar a leitura, esse exemplo está formatado com quebras de linha e não codificado de URL. Em seu código, a string de consulta deve ser uma string contínua codificada de URL.

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

Sigv4a

O exemplo a seguir mostra uma consulta para a ação `CreateBucket` usando o AWS SigV4a que contém as informações de autenticação. Para facilitar a leitura, esse exemplo está formatado com quebras de linha e não codificado de URL. Em seu código, a string de consulta deve ser uma string contínua codificada de URL.

```
https://ec2.amazonaws.com/?
Action=CreateBucket&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-ECDSA-P256-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/s3/aws4_request&
X-Amz-Region-Set=us-west-1&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date;x-amz-region-set&
X-Amz-Signature=calculated-signature
```

Exemplos de assinatura de solicitação

Os exemplos a seguir de solicitações de assinatura da AWS mostram como você pode usar o SigV4 para assinar solicitações enviadas sem o AWS SDK ou a ferramenta de linha de comando da AWS.

Upload do Amazon S3 baseado em navegador usando HTTP POST

Solicitações de autenticação: os uploads baseados em navegador descrevem a assinatura e as informações relevantes que o Amazon S3 usa para calcular a assinatura ao receber a solicitação.

Exemplo: o upload baseado em navegador usando HTTP POST (usando o AWS Signature Version 4) fornece mais informações com um exemplo de política POST e um formulário que você pode usar para fazer upload de um arquivo. A política de exemplo e as credenciais fictícias mostram o fluxo de trabalho e a assinatura e o hash da política resultantes.

Solicitações autenticadas do VPC Lattice

Os exemplos de solicitações autenticadas do Signature Version 4 (SigV4) fornecem exemplos em Python e Java que mostram como você pode realizar a assinatura de solicitações com e sem interceptores personalizados.

Usar o Signature Version 4 com o Amazon Translate

Live Translations in the Metaverse mostra como criar uma aplicação que produz uma solução de tradução quase em tempo real. Essa solução de tradução de fala para fala usa o SigV4 da AWS na codificação de fluxo de eventos para produzir transcrições em tempo real.

Usar o Signature Version 4 com o Neptune

Exemplo: conectar-se ao Neptune usando Python com a assinatura do Signature Version 4 mostra como fazer solicitações assinadas para o Neptune usando Python. Esse exemplo inclui variações para usar uma chave de acesso ou credenciais temporárias.

Assinatura de solicitações HTTP no Amazon Glacier

Exemplo de cálculo de assinatura para API de streaming explica os detalhes da criação de uma assinatura para o Upload Archive (arquivo POST), uma das duas APIs de streaming no Amazon Glacier.

Solicitações HTTP no Amazon SWF

Fazer solicitações HTTP no Amazon SWF mostra o conteúdo do cabeçalho de uma solicitação JSON no Amazon SWF.

Cálculo de assinatura para APIs de streaming no Amazon OpenSearch Service

Assinatura de uma solicitação de pesquisa do Amazon OpenSearch Service com AWS SDK para PHP versão 3 inclui um exemplo de como enviar solicitações HTTP assinadas para o Amazon OpenSearch Service.

Projetos de exemplo em repositório de amostras da AWS

Os projetos de exemplo a seguir mostram como assinar solicitações para fazer solicitações da API Rest para serviços da AWS com linguagens comuns, como Python, Node.js, Java, C#, Go e Rust.

Projetos do Signature Version 4a

O projeto sigv4-signing-examples fornece exemplos de como assinar solicitações com SigV4 para fazer solicitações da API Rest para Serviços da AWS com linguagens comuns, como Python, Node.js, Java, C#, Go e Rust.

O projeto sigv4a-signing-examples fornece exemplos para assinar solicitações de API multirregionais, por exemplo, Pontos de acesso multirregionais no Amazon S3.

Publicar no AWS IoT Core

O código Python para publicar no AWS IoT Core usando o protocolo HTTPs fornece orientação sobre como publicar mensagens no AWS IoT Core usando o protocolo Https e autenticação SigV4 da AWS. Tem duas implementações de referência, uma em Python e outra em NodeJS.

A aplicação .Net Framework para publicar no AWS IoT Core usando o protocolo HTTPs fornece orientação sobre como publicar mensagens no AWS IoT Core usando o protocolo HTTPS e autenticação SigV4 da AWS. Esse projeto também inclui uma implementação equivalente do .NET Core.

Solucionar problemas de assinatura do Signature Version 4 para solicitações de API da AWS

> **Important**
>
> A menos que você esteja usando AWS SDKs ou a CLI, é necessário escrever código para calcular assinaturas que forneçam informações de autenticação em suas solicitações. O cálculo da assinatura no SigV4 pode ser uma tarefa complexa. Por isso, recomenda-se usar os AWS SDKs ou a CLI sempre que possível.

Ao desenvolver um código que cria uma solicitação assinada, uma mensagem HTTP 403 `SignatureDoesNotMatch` poderá ser recebida dos Serviços da AWS. Esses erros significam que o valor da assinatura em sua solicitação HTTP para a AWS não correspondeu à assinatura calculada pelo AWS service (Serviço da AWS). Os erros HTTP 401 `Unauthorized` são retornados quando as permissões não permitem que o chamador faça a solicitação.

As solicitações de API podem retornar um erro se:

- A solicitação da API não é assinada e usa a autenticação do IAM.
- As credenciais do IAM usadas para assinar a solicitação estão incorretas ou não têm permissão para invocar a API.
- A assinatura da solicitação de API assinada não corresponde à assinatura calculada pelo serviço da AWS.
- O cabeçalho da solicitação da API está incorreto.

> **Note**
>
> Atualize seu protocolo de assinatura do AWS Signature Version 2 (SigV2) para o AWS Signature Version 4 (SigV4) antes de explorar outras soluções de erro. Serviços, como o Amazon S3, e regiões não oferecem mais suporte a assinaturas SigV2.

Possíveis causas

- Erros de credencial
- Erros de solicitação canônica e string de assinatura
- Erros de escopo de credenciais
- Erros de assinatura de chave

Erros de credencial

Certifique-se de que a solicitação de API seja assinada com o SigV4. Se a solicitação da API não estiver assinada, o seguinte erro poderá ser recebido: `Missing Authentication Token`. Adicione a assinatura que falta e reenvie a solicitação.

Verifique se as credenciais de autenticação para a chave de acesso e a chave secreta estão corretas. Se a chave de acesso estiver incorreta, o seguinte erro poderá ser recebido:

`Unauthorized`. Certifique-se de que a entidade usada para assinar a solicitação esteja autorizada a fazer a solicitação. Para obter detalhes, consulte Solucionar problemas de mensagens de erro de acesso negado.

Erros de solicitação canônica e string de assinatura

Se você calculou incorretamente a solicitação canônica em Criar um hash para a solicitação canônica ou Criar uma string para assinar, a etapa de verificação de assinatura executada pelo serviço falhará com a mensagem de erro:

```
The request signature we calculated does not match the signature you provided
```

Quando o serviço da AWS recebe uma solicitação assinada, ele recalcula a assinatura. Se houver diferenças nos valores, as assinaturas não coincidirão. Compare a solicitação canônica e a string à sua solicitação assinada com o valor na mensagem de erro. Modifique o processo de assinatura se houver alguma diferença.

> **Note**
>
> Você também pode verificar se não enviou a solicitação por meio de um proxy que modifica os cabeçalhos ou a solicitação.

Example Exemplo de solicitação canônica

```
GET                                                    -------- HTTP method
/                                                      -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                       -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                          -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com    -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                           -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501       -------- Hash
 of the payload
```

Para verificar se a chave secreta corresponde ao ID da chave de acesso, é possível testá-la com uma implementação em funcionamento conhecida. Por exemplo, use um AWS SDK ou a AWS CLI para fazer uma solicitação para a AWS.

Cabeçalho da solicitação de API

Quando o cabeçalho de autorização está vazio, a chave ou assinatura da credencial está ausente ou está incorreta, o cabeçalho não começa com um nome de algoritmo ou os pares de valores-chave não incluem um sinal de igual, você recebe um dos seguintes erros:

- O cabeçalho de autorização não pode estar vazio.
- O cabeçalho de autorização requer o parâmetro “Credential”.
- O cabeçalho de autorização requer o parâmetro “Signature”.
- A assinatura contém um par de chave=valor inválido (sem sinal de igualdade) no cabeçalho de autorização.

Certifique-se de que o cabeçalho de autorização SigV4 que você adicionou em Calcular a assinatura inclua a chave de credencial correta e também a data da solicitação usando HTTP Date ou o cabeçalho.

Se você recebeu um erro IncompleteSignatureException e a construção da assinatura está correta, você pode verificar se o cabeçalho de autorização não foi modificado em trânsito para o computando um hash SHA-256 e a codificação B64 do cabeçalho de autorização em sua solicitação AWS service (Serviço da AWS) do lado do cliente.

1. Obtenha o cabeçalho de autorização que você enviou na solicitação. Seu cabeçalho de autorização é semelhante ao seguinte exemplo:

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

2. Calcule um hash SHA-256 do cabeçalho de autorização.

```
hashSHA256(rawAuthorizationHeader) = hashedAuthorizationHeader
```

3. Codifique o cabeçalho de autorização com hash no formato Base64.

```
base64(hashedAuthorizationHeader) = encodedHashedAuthorizationHeader
```

4. Compare a string codificada e com hash que você acabou de calcular com a string que você recebeu na sua mensagem de erro. Sua mensagem de erro deve ser semelhante ao exemplo a seguir:

```
com.amazon.coral.service#IncompleteSignatureException:
The signature contains an in-valid key=value pair (missing equal-sign)
in Authorization header (hashed with SHA-256 and encoded with Base64):
'9c574f83b4b950926da4a99c2b43418b3db8d97d571b5e18dd0e4f3c3ed1ed2c'.
```

- Se os dois hashes forem diferentes, então alguma parte do cabeçalho de autorização foi alterada em trânsito. Essa alteração pode ocorrer devido ao fato de seus manipuladores de rede ou cliente anexarem cabeçalhos assinados ou alterarem o cabeçalho de autorização de alguma forma.
- Se os dois hashes corresponderem, o cabeçalho de autorização que você enviou na solicitação corresponderá ao AWS recebido. Analise a mensagem de erro que você recebeu para determinar se o problema é resultado de credenciais ou assinaturas incorretas. Esses erros são abordados nas outras seções desta página.

Erros de escopo de credenciais

O escopo da credencial criado em Criar uma string para assinar restringe a assinatura a uma data, uma região e um serviço específicos. Essa string tem o seguinte formato:

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Se você estiver usando o SIGv4a, a região não estará incluída no escopo da credencial.

Data

Se o escopo da credencial não especificar a mesma data que o cabeçalho x-amz-date, a etapa de verificação da assinatura falhará com a seguinte mensagem de erro:

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Se a solicitação especificar um horário no futuro, a etapa de verificação da assinatura falhará com a seguinte mensagem de erro:

```
Signature not yet current: date is still later than date
```

Se a solicitação tiver expirado, a etapa de verificação da assinatura falhará com a seguinte mensagem de erro:

```
Signature expired: date is now earlier than date
```

Região

Se o escopo da credencial não especificar a mesma região que a solicitação, a etapa de verificação da assinatura falhará com a seguinte mensagem de erro:

```
Credential should be scoped to a valid Region, not region-code
```

Serviço

Se o escopo da credencial não especificar o mesmo serviço que o cabeçalho host, a etapa de verificação da assinatura falhará com a seguinte mensagem de erro:

```
Credential should be scoped to correct service: 'service'
```

String de término

Se o escopo da credencial não terminar com aws4_request, a etapa de verificação da assinatura falhará com a seguinte mensagem de erro:

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Erros de assinatura de chave

Erros causados por derivação incorreta da chave de assinatura ou uso incorreto de criptografia são mais difíceis de solucionar. Após verificar se a string canônica e a string a ser assinada estão corretas, você também pode verificar se há um dos seguintes problemas:

- A chave de acesso secreta não corresponde à ID de chave de acesso especificada.
- Existe um problema com seu código de derivação de chaves.

Para verificar se a chave secreta corresponde ao ID da chave de acesso, é possível testá-la com uma implementação em funcionamento conhecida. Por exemplo, use um AWS SDK ou a AWS CLI para fazer uma solicitação para a AWS. Para obter exemplos, consulte Exemplos de assinatura de solicitação

Referência de política JSON do IAM

Esta seção apresenta sintaxe detalhada, descrições e exemplos dos elementos, variáveis e lógica de avaliação de políticas de JSON no IAM. Para obter mais informações gerais, consulte Visão geral das políticas de JSON.

Esta referência inclui as seções a seguir.

- Referência de elemento de política JSON do IAM: saiba mais sobre os elementos que você pode usar ao criar uma política. Veja exemplos adicionais de políticas e saiba mais sobre condições, tipos de dados suportados e como eles são usados em vários serviços.
- Lógica da avaliação de política – essa seção descreve solicitações da AWS, como elas são autenticadas e como a AWS usa políticas para determinar o acesso a recursos.
- Gramática da linguagem das políticas de JSON do IAM: essa seção apresenta uma gramática formal para a linguagem usada para criar políticas no IAM.
- AWSPolíticas gerenciadas pela para funções de trabalho – essa seção lista todas as políticas gerenciadas da AWS que são mapeadas diretamente para funções de trabalho comuns no setor de TI. Use essas políticas para facilmente conceder as permissões necessárias para executar as tarefas esperadas de alguém com determinada função de trabalho. Essas políticas consolidam permissões para vários serviços em uma única política.
- AWSChaves de contexto de condições globais da: esta seção inclui uma lista de todas as chaves de condição globais da AWS que você pode usar para limitar permissões em uma política do IAM.
- Chaves de contexto de condição do IAM e do AWS STS: esta seção inclui uma lista de todas as chaves de condição do IAM e do AWS STS que você pode usar para limitar permissões em uma política do IAM.
- Ações, recursos e chaves de condição para produtos da AWSServiços: esta seção apresenta uma lista de todas as operações de API da AWS que você pode usar como permissões em uma política

do IAM. Também inclui as chaves de condição específicas que podem ser usadas para refinar a solicitação.

Referência de elemento de política JSON do IAM

Documentos de política JSON são compostos de elementos. Os elementos são listados aqui na ordem geral em que são usados em uma política. A ordem dos elementos não importa, por exemplo, o elemento `Resource` pode vir antes do elemento `Action`. Não é necessário especificar todos os elementos `Condition` da política. Para saber mais sobre a estrutura e a finalidade gerais de um documento de política JSON, consulte Visão geral das políticas de JSON.

Alguns elementos de política JSON são mutuamente exclusivos. Isso significa que você não pode criar uma política que usa ambos. Por exemplo, você não pode usar `Action` e `NotAction` na mesma instrução de política. Outros pares que são mutuamente exclusivos incluem `Principal`/`NotPrincipal` e `Resource`/`NotResource`.

Os detalhes do que entra em uma política variam para cada serviço, dependendo de quais ações o serviço disponibiliza, que tipos de recursos ele contém e assim por diante. Quando você estiver gravando políticas para um serviço específico, é útil ver exemplos de políticas para esse serviço. Para obter uma lista de todos os serviços compatíveis com o IAM e links para a documentação desses serviços que discute o IAM e políticas, consulte AWSServiços da que funcionam com o IAM.

Quando você cria ou edita uma política JSON, o IAM pode executar a validação de políticas para ajudar você a criar uma política eficaz. O IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de política do IAM. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer.

Tópicos

- Elementos de política JSON do IAM: Version
- Elementos de política JSON do IAM: Id
- Elementos de política JSON do IAM: Statement
- Elementos de política JSON do IAM: Sid
- Elementos de política JSON do IAM: Effect
- AWS Elementos da política JSON da: Principal
- Elementos da política JSON da AWS: NotPrincipal

- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: NotAction
- Elementos de política JSON do IAM: Resource
- Elementos de política JSON do IAM: NotResource
- Elementos de política JSON do IAM: Condition
- Elementos de política do IAM: variáveis e etiquetas
- Elementos de política JSON do IAM: tipos de dados compatíveis

Elementos de política JSON do IAM: Version

> **Nota de desambiguação**
>
> O elemento de política JSON `Version` é diferente de uma versão de política. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. A versão da política, por outro lado, é criada quando você faz alterações em uma política gerenciada pelo cliente no IAM. A política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. Se você estava procurando informações sobre o suporte a várias versões disponível para políticas gerenciadas, consulte the section called “Versionamento de políticas do IAM”.

O elemento de política `Version` especifica as regras de sintaxe de linguagem que devem ser usadas para processar uma política. Para usar todos os recursos de política disponíveis, inclua o elemento `Version` a seguir fora do elemento `Statement` em todas as suas políticas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
```

```
}
```

O IAM oferece suporte aos seguintes valores do elemento `Version`:

- `2012-10-17`. Esta é a versão atual da linguagem da política e você deve sempre incluir um elemento `Version` e defini-lo como `2012-10-17`. Caso contrário, você não poderá usar recursos, como variáveis de política, que foram introduzidos com esta versão.
- `2008-10-17`. Esta é uma versão anterior da linguagem da política. Talvez você veja essa versão em antigas políticas existentes. Não use essa versão para as políticas novas ou quando você atualizar alguma política existente. Recursos mais recentes, como variáveis de política, não funcionarão com a sua política. Por exemplo, as variáveis como `${aws:username}` não serão reconhecidas como variáveis e serão tratadas como strings literais na política.

Elementos de política JSON do IAM: Id

O elemento `Id` especifica um identificador opcional para a política. O ID é usado de forma diferente em diferentes serviços. O ID é permitido em políticas baseadas em recursos, mas não em políticas baseadas em identidades.

Para serviços que permitem que você defina um elemento `ID`, recomendamos que você use um UUID (GUID) para o valor ou que incorpore um UUID como parte do ID para garantir a exclusividade.

JSON

```
{
  "Version":"2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Alguns produtos da AWS (por exemplo, Amazon SQS ou Amazon SNS) podem exigir este elemento e possuem requisitos de exclusividade para ele. Para obter informações específicas do serviço sobre a gravação de políticas, consulte a documentação para o serviço com o qual você está trabalhando.

Elementos de política JSON do IAM: Statement

O elemento `Statement` é o principal elemento de uma política. Este elemento é obrigatório. O elemento `Statement` pode conter uma única instrução ou uma matriz de instruções individuais. Cada bloco de instrução individual deve ser colocado entre chaves { }. Para várias instruções, a matriz deve estar entre colchetes [ ].

```
"Statement": [{...},{...},{...}]
```

O exemplo a seguir mostra uma política que contém uma matriz de três instruções dentro de um único elemento `Statement`. (A política permite que você acesse sua própria “pasta base” no console do Amazon S3.) A política inclui a variável `aws:username`, que é substituída durante a avaliação da política pelo nome de usuário da solicitação. Para obter mais informações, consulte Introdução.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
```

```
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

Elementos de política JSON do IAM: Sid

É possível fornecer um `Sid` (ID de instrução) como identificador opcional para a instrução da política. Você pode atribuir um valor `Sid` a cada instrução em uma matriz de instruções. Você pode usar o valor `Sid` como uma descrição para a instrução de política. Em serviços que permitem que você especifique um elemento `ID`, como o SQS e o SNS, o valor `Sid` é apenas um subID do ID do documento de política. No IAM, o valor `Sid` deve ser exclusivo em uma política JSON.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

O elemento `Sid` é compatível com letras maiúsculas ASCII (A-Z), letras minúsculas (a-z) e números (0-9).

O IAM não expõe o `Sid` na API do IAM. Você não pode recuperar uma instrução específica com base neste ID.

> **Note**
>
> Alguns produtos da AWS (por exemplo, Amazon SQS ou Amazon SNS) podem exigir este elemento e possuem requisitos de exclusividade para ele. Para obter informações específicas do serviço sobre a gravação de políticas, consulte a documentação do serviço com o qual você está trabalhando.

Elementos de política JSON do IAM: Effect

O elemento `Effect` é obrigatório e especifica se a instrução resulta em uma permissão ou uma negação explícita. Os valores válidos para `Effect` são `Allow` e `Deny`. O valor `Effect` diferencia maiúsculas de minúsculas.

```
"Effect":"Allow"
```

Por padrão, o acesso aos recursos é negado. Para permitir o acesso a um recurso, você deve definir o elemento `Effect` como `Allow`. Para substituir uma permissão (por exemplo, para substituir uma permissão que está em vigor), você define o elemento `Effect` como `Deny`. Para obter mais informações, consulte Lógica da avaliação de política.

AWS Elementos da política JSON da: Principal

Use o elemento `Principal` em uma política JSON baseada em recursos para especificar a entidade principal cujo acesso a um recurso é permitido ou negado.

Você pode usar o elemento `Principal` em políticas baseadas em recursos. Vários serviços suportam políticas baseadas em recursos, inclusive o IAM. O tipo de política do IAM baseada em recursos é uma política de confiança em função. Nas funções do IAM, use o elemento `Principal` na política de confiança em função para especificar quem pode assumir a função. Para o acesso entre contas, você deve especificar o identificador de 12 dígitos da conta confiável. Para saber se as entidades de contas fora de sua zona de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?.

> **Note**
>
> Depois de criar a função, você pode alterar a conta para "*" a fim de permitir que todas as pessoas assumam a função. Se você fizer isso, recomendamos que você limite quem pode acessar a função através de outros meios, tal como um elemento `Condition` que limita o acesso somente a determinados endereços IP. Não deixe a função acessível a todos!

Outros exemplos de recursos que dão suporte a políticas baseadas em recursos incluem um bucket do Amazon S3 ou uma AWS KMS key.

Você não pode usar o elemento `Principal` em uma política baseada em identidade. As políticas baseadas em identidade são políticas de permissões que você anexa a identidades do IAM (usuários, grupos ou funções). Nesses casos, a entidade principal é implicitamente a identidade à qual a política está anexada.

Tópicos

- Como especificar uma entidade principal
- Conta da AWSEntidades principais da
- Entidades de segurança de função do IAM
- Entidades principais da sessão de função
- Entidades principais federadas OIDC
- Entidades principais federadas SAML
- Entidades principais de usuário do IAM
- Entidades principais do Centro de identidade do IAM
- AWS STSEntidades principais federadas do
- AWSResponsáveis pelos serviços da
- AWSEntidades principais de serviço da nas regiões de aceitação
- Todas as entidades principais
- Mais informações

Como especificar uma entidade principal

Você especifica uma entidade principal no elemento `Principal` de uma política baseada em recursos ou em chaves de condição que dão suporte a entidades principais.

Você pode especificar qualquer um dos seguintes principais em uma política:

- Conta da AWS e usuário raiz
- Perfis do IAM
- Sessões de função
- Usuários do IAM
- Entidades principais federadas
- AWSServiços da
- Todas as entidades principais

Não é possível identificar um grupo de usuários como entidade principal em uma política (como uma política baseada em recursos) porque os grupos são relacionados com permissões, não autenticação, e as entidades principais são entidades autenticadas do IAM.

Você pode especificar mais de uma entidade de segurança para cada um dos tipos de entidade de segurança nas seções a seguir usando uma matriz. As matrizes podem levar um ou mais valores. Ao especificar mais de uma entidade principal em um elemento, você concede permissões para cada entidade principal. Isso é um `OR` lógico e não um `AND` lógico, porque você autentica como uma entidade principal por vez. Se incluir mais de um valor, use colchetes (`[` e `]`) e delimite com vírgulas cada entrada da matriz. A política de exemplo a seguir define permissões para a conta 123456789012 ou para a conta 555555555555.

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> Não é possível usar um curinga para fazer a correspondência de parte de um nome de entidade principal ou ARN.

Conta da AWSEntidades principais da

Você pode especificar identificadores de Conta da AWS no elemento `Principal` de uma política baseada em recursos ou em chaves de condição que oferecem suporte a entidades principais. Isso delega autoridade para a conta. Quando você permite o acesso a uma conta diferente, um administrador nessa conta deve então conceder acesso a uma identidade (usuário ou função do IAM) nessa conta. Ao especificar uma Conta da AWS, você pode usar o ARN da conta (arn:aws:iam::*account-ID*:root), ou uma forma abreviada que consiste no prefixo `"AWS":` seguido pelo ID da conta.

Por exemplo, dado um ID da conta de `123456789012`, você pode usar um dos seguintes métodos para especificar essa conta no elemento `Principal`:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

O ARN da conta e o ID da conta abreviado têm comportamento semelhante. Ambos delegam permissões à conta. O uso do ARN da conta no elemento `Principal` não limita permissões apenas para o usuário raiz da conta.

> **Note**
>
> Quando você salva uma política baseada em recursos que inclui o ID abreviado da conta, o serviço pode convertê-lo no ARN da entidade principal. Isso não altera a funcionalidade da política.

Alguns serviços da AWS são compatíveis com opções adicionais para especificar uma conta do principal. Por exemplo, o Amazon S3 permite que você especifique um ID de usuário canônico usando o seguinte formato:

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Você também pode especificar mais de uma Conta da AWS (ou ID de usuário canônico) como uma entidade principal usando uma matriz. Por exemplo, você pode especificar uma entidade principal em uma política de bucket usando todos os três métodos.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

Entidades de segurança de função do IAM

Você pode especificar ARNs de entidades principais de funções do IAM no elemento `Principal` de uma política baseada em recursos ou em chaves de condição que suportam entidades principais. As funções do IAM são identidades. No IAM, identidades são recursos aos quais você pode atribuir permissões. Funções confiam em outra identidade autenticada para assumir a respectiva função. Isso inclui uma entidade principal na AWS ou um usuário de um provedor de identidade externo (IdP). Quando uma entidade principal ou uma identidade assume uma função, ela recebe credenciais de segurança temporárias com as permissões da função assumida. Quando elas usam essas credenciais de sessão para executar operações na AWS, elas se tornam uma entidade principal da sessão de função.

Quando você especifica uma entidade principal de função em uma política baseada em recursos, as permissões efetivas para a entidade principal são limitadas por qualquer tipo de política que limite as permissões para a função. Isso inclui políticas de sessão e limites de permissões. Para mais informações sobre como as permissões efetivas de uma sessão de função são avaliadas, consulte Lógica da avaliação de política.

Para especificar o ARN da função no elemento `Principal`, use o seguinte formato:

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Se o seu elemento `Principal` em uma política de confiança de perfil contiver um ARN que aponte para uma um perfil do IAM específico, esse ARN será transformado no ID exclusivo da entidade principal do perfil quando você salvar a política. Isso ajuda a reduzir o risco de alguém elevar seus privilégios ao remover e recriar a função. Normalmente, você não vê esse ID no console, porque o IAM utiliza uma transformação reversa de volta para o ARN de função quando a política de confiança é exibida.

No entanto, se você excluir a função, a relação é interrompida. A política não se aplica mais, mesmo se você recriar a função, pois a nova função possui um novo ID principal que não corresponde ao ID armazenado na política de confiança. Quando isso acontece, o ID da entidade principal aparece nas políticas baseadas em recursos porque a AWS não pode mais mapeá-lo de volta para um ARN válido.
O resultado é que, se você excluir e recriar um perfil referenciado no elemento `Principal` de uma política de confiança, deverá editar o perfil na política para substituir o ID da entidade principal pelo ARN correto. O ARN se transforma novamente no novo ID da entidade principal da função quando você salva a política. Para obter mais informações, consulte Understanding AWS's Handling of Deleted IAM roles in Policies.

Como alternativa, você pode especificar a entidade principal da função como a entidade principal em uma política baseada em recurso ou criar uma política de permissão ampla que use a chave de condição `aws:PrincipalArn`. Quando você usa essa chave, a entidade principal de sessão de função recebe as permissões com base no ARN da função assumida e não no ARN da sessão resultante. Como a AWS não converte ARNs da chave de condição em IDs, as permissões concedidas ao ARN da função persistirão se você excluir a função e criar uma nova função com o mesmo nome. Tipos de políticas baseadas em identidade, como limites de permissões ou políticas de sessão, não limitam as permissões concedidas usando a chave de condição `aws:PrincipalArn` com um curinga (*) no elemento `Principal`, a menos que as políticas baseadas em identidade contenham uma negação explícita.

Entidades principais da sessão de função

Você pode especificar sessões de função no elemento `Principal` de uma política baseada em recursos ou em chaves de condição que suportam entidades principais. Quando uma entidade principal ou uma identidade assume uma função, ela recebe credenciais de segurança temporárias com as permissões da função assumida. Quando elas usam essas credenciais de sessão para executar operações na AWS, elas se tornam uma entidade principal da sessão de função.

O formato que você usa para uma entidade principal da sessão de função depende da operação AWS STS que foi usada para assumir a função.

> **Important**
>
> A AWS recomenda usar entidades principais de perfil do IAM em suas políticas, em vez de entidades principais de sessão de perfil, sempre que possível. Use instruções `Condition` e chaves de condição para reduzir ainda mais o acesso quando necessário.

Para especificar o ARN da entidade principal da sessão de perfil assumida no elemento `Principal`, use o seguinte formato:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Além disso, os administradores podem projetar um processo para controlar como as sessões de função são emitidas. Por exemplo, eles podem fornecer uma solução de um clique para seus usuários que cria um nome de sessão previsível. Se o administrador fizer isso, você poderá usar as entidades principais da sessão de função em suas políticas ou chaves de condição. Caso contrário, você pode especificar o ARN do perfil como a entidade principal na chave de condição do `aws:PrincipalArn`. O modo como você especifica a função como entidade principal pode alterar as permissões efetivas para a sessão resultante. Para obter mais informações, consulte Entidades de segurança de função do IAM.

Entidades principais federadas OIDC

Uma entidade principal federada do OIDC é a entidade principal usada ao chamar a API `AssumeRoleWithWebIdentity` do AWS STS com um token web JSON (JWT) de um IDP compatível com o OIDC, também conhecido como OpenID Provider (OP), para solicitar credenciais temporárias da AWS. Uma entidade principal federada do OIDC pode representar um IDP do OIDC em sua conta da AWS, ou os 4 provedores de identidade integrados: Login with Amazon, Google, Facebook, e Amazon Cognito.

Usuários, workloads ou sistemas que receberam um JWT de seu IDP do OIDC podem chamar `AssumeRoleWithWebIdentity` usando o JWT para solicitar credenciais de segurança temporárias da AWS para um perfil do IAM configurado para confiar no IDP do OIDC que emitiu o JWT. O JWT pode ser um token de identificação, token de acesso ou um token JWT entregue por qualquer outro método, desde que atenda aos requisitos listados pelo AWS STS. Para obter mais informações, consulte Cenários comuns e Solicitar credenciais por meio de um provedor OIDC.

Use este tipo de entidade principal na política de confiança de seu perfil para permitir ou negar permissões de chamada `AssumeRoleWIthWebIdentity` usando um IDP do OIDC que existe ne sua Conta da AWS ou em um dos quatro IDPs integrados. Para especificar o ARN da entidade principal federada ao OIDC no elemento `Principal` de uma política de confiança de perfil, use um dos quatro formatos a seguir para os IDPs OIDC integrados:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Ao usar um provedor OIDC que você adiciona à sua conta, como o GitHub, você especifica o ARN do provedor na política de confiança da sua função. Essa configuração a você escrever políticas do IAM que controlam o acesso especificamente para usuários autenticados por meio de seu provedor de identidade personalizado.

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/full-OIDC-identity-provider-URL" }
```

Por exemplo, se GitHub for o provedor de identidade da Web confiável, o ARN da sessão do perfil do OIDC no elemento da `Principal` de uma política de confiança de perfil usará o seguinte formato:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:oidc-provider/
tokens.actions.githubusercontent.com" }
```

Consulte Configurar o OpenID Connect na Amazon Web Services para obter mais informações.

As entidades principais federadas do OIDC não oferecem suporte a outros tipos de políticas além das políticas de confiança de funções.

Entidades principais federadas SAML

Uma entidade principal federada do SAML é a entidade principal usada ao chamar a API `AssumeRoleWithSAML` do AWS STS para solicitar credenciais temporárias da AWS usando uma

declaração do SAML. Você pode usar seu provedor de identidades (IdP) SAML para se conectar e assumir um perfil do IAM usando essa operação. Semelhante a `AssumeRoleWithWebIdentity`, `AssumeRoleWithSAML` não exige credenciais da AWS para autenticação. Em vez disso, primeiro, os usuários se autenticam com seu provedor de identidade SAML e fazem a chamada de API `AssumeRoleWithSAML` usando sua declaração SAML ou são redirecionados para a página de login/SAML da AWS para entrar no Console de gerenciamento da AWS. Para mais informações sobre quais entidades principais podem assumir uma função usando essa operação, consulte Compare credenciais do AWS STS.

Use este tipo de entidade principal em sua política de confiança de perfil para permitir ou negar permissões com base no provedor de identidade SAML confiável. Para especificar o ARN da sessão de perfil de identidade SAML no elemento `Principal` de uma política de confiança de perfil, use o seguinte formato:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

Entidades principais de usuário do IAM

Você pode especificar usuários do IAM no elemento `Principal` de uma política baseada em recursos ou em chaves de condições que suportam entidades principais.

> **Note**
>
> Em um elemento `Principal`, a parte do nome de usuário do nome do recurso da Amazon (ARN) diferencia maiúsculas e minúsculas.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Ao especificar usuários em um elemento `Principal`, você não pode usar um curinga (*) para se referir a "todos os usuários". Entidades principais sempre devem nomear usuários específicos.

> **Important**
>
> Se o seu elemento `Principal` em uma política de confiança de função contiver um ARN que aponte para um usuário específico do IAM, o IAM transformará o ARN no ID exclusivo da entidade principal do usuário quando você salvar a política. Isso ajuda a reduzir o risco de alguém elevar seus privilégios ao remover e recriar o usuário. Normalmente, você não vê esse ID no console, pois há também uma transformação reversa de volta para o ARN do usuário quando a política de confiança é exibida.
> No entanto, se você excluir o usuário, a relação é interrompida. A política não se aplica mais, mesmo se você recriar o usuário. Isso porque o novo usuário tem um novo ID principal que não corresponde ao ID armazenado na política de confiança. Quando isso acontece, o ID da entidade principal aparece nas políticas baseadas em recursos porque a AWS não pode mais mapeá-lo de volta para um ARN válido.
> Como resultado, se você excluir e recriar um usuário mencionado no elemento `Principal` de uma política de confiança, você deve editar a função para substituir o agora incorreto ID da entidade principal pelo ARN correto. O IAM transformará novamente o ARN no novo ID da entidade principal do usuário quando você salvar a política.

Entidades principais do Centro de identidade do IAM

No Centro de identidade do IAM, a entidade principal em uma política baseada em recursos deve ser definida como a entidade principal da Conta da AWS. Para especificar o acesso, faça referência ao ARN do perfil do conjunto de permissões no bloco de condições. Para obter detalhes, consulte Referencing permission sets in resource policies, Amazon EKS, and AWS KMS no Guia do usuário do Centro de identidade do IAM.

AWS STSEntidades principais federadas do

Você pode especificar sessões de usuário federado no elemento `Principal` de uma política baseada em recursos ou em chaves de condição que suportam entidades principais.

> **Important**
>
> A AWS recomenda que você limite o uso de sessões do usuário federado do AWS STS. Em vez disso, use os perfis do IAM.

Uma entidade principal de usuário federado do AWS STS é criado por meio da operação chamada `GetFederationToken` com credenciais do IAM de longa duração. As permissões do usuário federado são a interseção da entidade principal que chamou `GetFederationToken` e das políticas de sessão passadas como parâmetros para a API `GetFederationToken`.

Na AWS, um Usuário raiz da conta da AWS ou usuários do IAM podem fazer autenticação usando chaves de acesso de longo prazo. Para mais informações sobre quais entidades principais podem fazer a federação usando essa operação, consulte Compare credenciais do AWS STS.

- Usuário federado do IAM: um usuário do IAM se federa usando a operação `GetFederationToken`, que resulta em uma sessão de usuário federado para esse usuário do IAM.
- Usuário-raiz federado: um usuário-raiz se federa usando a operação `GetFederationToken`, o que resulta em uma sessão de usuário federado para esse usuário-raiz.

Quando um usuário do IAM ou usuário raiz solicita credenciais temporárias do AWS STS usando essa operação, ele inicia uma sessão temporária de usuário federado. O ARN dessa sessão é baseado na identidade original que foi federada.

Para especificar o ARN da sessão de usuário federado no elemento `Principal`, use o seguinte formato:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWSResponsáveis pelos serviços da

Você pode especificar serviços da AWS no elemento `Principal` de uma política baseada em recursos ou em chaves de condição que suportam entidades principais. Uma entidade principal de serviço é um identificador para um serviço.

As funções do IAM que podem ser assumidas por um produto da AWS são chamadas de funções de serviço. As funções de serviço devem incluir uma política de confiança. Políticas de confiança são políticas baseadas em recursos anexadas a uma função que define quais entidades principais podem assumir a função. Algumas funções de serviço têm políticas de confiança predefinidas. No entanto, em alguns casos, você deve especificar o escopo principal do serviço na política de confiança. A entidade principal do serviço em uma política do IAM não pode ser `"Service": "*"`.

> **Important**
>
> O identificador de uma entidade principal de serviço inclui o nome do serviço e geralmente está no seguinte formato:
> *service-name*.amazonaws.com

O escopo principal do serviço é definido pelo serviço. É possível encontrar a entidade principal de serviço de alguns serviços abrindo AWSServiços da que funcionam com o IAM, verificando se o serviço tem um Sim na coluna Perfil vinculado ao serviço e abrindo o link de Sim para visualizar a documentação do perfil vinculado a esse serviço. Localize a seção Service-Linked Role Permissions (Permissões da função vinculada ao serviço) desse serviço para visualizar o principal do serviço.

O exemplo a seguir mostra uma política que pode ser anexada a uma função do serviço. A política permite que dois serviços, o Amazon ECS eo Elastic Load Balancing, assumam a função. Os serviços podem então realizar qualquer tarefa concedida pela política de permissões atribuída à função (não exibida). Para especificar vários principais de serviço, você não especifica dois elementos `Service`; pode ter apenas um. Em vez disso, você usa uma variedade de principais de serviços múltiplas como o valor de um único elemento `Service`.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWSEntidades principais de serviço da nas regiões de aceitação

Você pode lançar recursos em várias regiões da AWS e em algumas dessas regiões pelas quais você deve optar. Para obter uma lista completa das regiões pelas quais você deve optar, consulte Como gerenciar regiões da AWS no guia Referência geral da AWS.

Quando um serviço da AWS em uma região de aceitação faz uma solicitação dentro da mesma região, o formato do nome da entidade principal do serviço é identificado como a versão não regionalizada desse nome:

service-name.amazonaws.com

Quando um serviço da AWS em uma região de aceitação faz uma solicitação entre regiões para outra região, o formato do nome da entidade principal do serviço é identificado como a versão regionalizada desse nome:

service-name.*{region}*.amazonaws.com

Por exemplo, você tem um tópico do Amazon SNS localizado na região `ap-southeast-1` e um bucket do Amazon S3 localizado na região de aceitação `ap-east-1`. Você deseja configurar as notificações do bucket do S3 para publicar mensagens no tópico do SNS. Para permitir que o serviço S3 publique mensagens no tópico do SNS, você deve conceder à entidade principal do serviço S3 a permissão `sns:Publish` por meio da política de acesso baseada em recursos do tópico.

Se você especificar a versão não regionalizada da entidade principal do serviço S3, `s3.amazonaws.com`, na política de acesso do tópico, a solicitação `sns:Publish` do bucket para o tópico falhará. O exemplo a seguir especifica a entidade principal do serviço S3 não regionalizado no elemento de política `Principal` da política de acesso a tópicos do SNS.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Como o bucket está localizado em uma região de aceitação e a solicitação é feita fora dessa mesma região, a entidade principal do serviço S3 aparece como o nome da entidade principal do serviço regionalizado, `s3.ap-east-1.amazonaws.com`. Você deve usar o nome da entidade principal do serviço regionalizado quando um serviço da AWS em uma região de aceitação fizer uma solicitação para outra região. Depois de especificar o nome da entidade principal do serviço regionalizado, se o bucket fizer uma solicitação `sns:Publish` ao tópico do SNS localizado em outra região, a solicitação será bem-sucedida. O exemplo a seguir especifica a entidade principal do serviço regionalizado S3 no elemento de política `Principal` da política de acesso a tópicos do SNS.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Políticas de recursos ou listas de permissões baseadas em entidades principais de serviços para solicitações entre regiões de uma região de aceitação para outra região só serão bem-sucedidas se você especificar o nome da entidade principal do serviço regionalizado.

> **Note**
>
> Para políticas de confiança de perfis do IAM, recomendamos usar o nome da entidade principal do serviço não regionalizado. Os recursos do IAM são globais e, portanto, o mesmo perfil pode ser usado em qualquer região.

Todas as entidades principais

É possível usar um curinga (*) para especificar todas as entidades principais no elemento `Principal` de uma política baseada em recursos ou em chaves de condições compatíveis com entidades principais. Políticas baseadas em recursos concedem permissões e chaves de condições são usadas para limitar as condições de uma instrução de política.

> **Important**
>
> É muito recomendável que você não use um caractere curinga (*) no elemento `Principal` de uma política baseada em recursos com um efeito `Allow`, a menos que pretenda conceder acesso público ou anônimo. Caso contrário, especifique entidades principais, serviços ou contas da AWS no elemento `Principal` e depois restrinja ainda mais o acesso no elemento `Condition`. Isso é especialmente verdadeiro para políticas de confiança do perfil do IAM, porque permitem que outras entidades principais se tornem uma entidade principal em sua conta.

Para políticas baseadas em recursos, usar um caractere curinga (*) com um efeito `Allow` concede acesso a todos os usuários, incluindo usuários anônimos (acesso público). Para usuários do IAM e entidades principais de função dentro da sua conta, nenhuma outra permissão é necessária. Para entidades principais em outras contas, elas também devem ter permissões baseadas em identidade em suas contas que lhes permitam acessar seu recurso. Isso é chamado de acesso entre contas.

Para usuários anônimos, os seguintes elementos são equivalentes:

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Não é possível usar um curinga para fazer a correspondência de parte de um nome de entidade principal ou ARN.

O exemplo a seguir mostra uma política baseada em recurso que pode ser usada em vez de Elementos da política JSON da AWS: NotPrincipal para negar explicitamente todas as entidades principais exceto as especificadas no elemento `Condition`. Essa política deve ser adicionada a um bucket do Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

Mais informações

Para saber mais, consulte:

- Exemplos de políticas de bucket no Guia do usuário do Amazon Simple Storage Service
- Exemplos de políticas do Amazon SNS no Guia do desenvolvedor do Amazon Simple Notification Service
- Exemplos de políticas do Amazon SQS no Guia do desenvolvedor do Amazon Simple Queue Service

- Políticas de chaves no Guia do desenvolvedor do AWS Key Management Service
- Identificadores de conta no Referência geral da AWS
- Federação OIDC

Elementos da política JSON da AWS: NotPrincipal

O elemento `NotPrincipal` usa `"Effect":"Deny"` para negar acesso a todas as entidades principais, exceto a entidade principal especificada no elemento `NotPrincipal`. Uma entidade principal pode ser um usuário do IAM, uma entidade principal de usuário federado do AWS STS, um perfil do IAM, uma sessão de perfil assumido, uma Conta da AWS, um serviço da AWS ou outro tipo de entidade principal. Para obter mais informações, sobre entidades principais, consulte AWS Elementos da política JSON da: Principal.

`NotPrincipal` deve ser usado com `"Effect":"Deny"`. O uso de `"Effect":"Allow"` não é compatível.

> **Important**
>
> Não recomendamos o uso de `NotPrincipal` para novas políticas baseadas em recursos como parte de sua estratégia de segurança e autorização. Quando você usa `NotPrincipal`, pode ser difícil solucionar os problemas causados por vários tipos de política. Em vez disso, recomendamos usar a chave de contexto `aws:PrincipalArn` com operadores de condição de ARN.

Principais pontos

- O elemento `NotPrincipal` é aceito em políticas baseadas em recursos para alguns serviços da AWS, incluindo endpoints da VPC. As políticas baseadas em recursos são políticas que você incorpora diretamente em um recurso. Não é possível usar o elemento `NotPrincipal` em uma política baseada em identidade do IAM ou em uma política de perfil do IAM de confiança.
- Não use declarações de política baseadas em recursos que incluam um elemento de política `NotPrincipal` com um efeito `Deny` para usuários ou perfis do IAM que tenham uma política de limite de permissões anexada. O elemento `NotPrincipal` com um efeito `Deny` sempre negará qualquer entidade principal do IAM que tenha uma política de limite de permissões anexada, independentemente dos valores especificados no elemento `NotPrincipal`. Isso faz com que alguns usuários ou perfis do IAM que, de outra forma, teriam acesso ao recurso, percam o acesso.

Recomendamos alterar suas declarações de política baseadas em recursos para usar o operador de condição `ArnNotEquals` com a chave de contexto `aws:PrincipalArn` para limitar o acesso, em vez do elemento `NotPrincipal`. Para obter mais informações sobre esses limites de permissões, consulte Limites de permissões para entidades do IAM.

- Ao usar `NotPrincipal`, você também deve especificar o ARN da entidade principal não negada. Caso contrário, a política pode negar o acesso à conta por completo contendo o principal. Dependendo do serviço que você incluir em sua política, a AWS pode validar primeiro a conta e, em seguida, o usuário. Se um usuário de função assumida (alguém que esteja usando uma função) está sendo avaliado, AWS pode validar primeiro a conta, em seguida, a função e, por fim, o usuário de função assumida. O usuário de função assumida é identificado pelo nome de sessão da função, especificado quando ele assumiu a função. Portanto, é recomendável incluir explicitamente o ARN para uma conta de usuário, ou incluir o ARN para a função e o ARN para a conta que contém essa função.
- O elemento `NotPrincipal` não é compatível com políticas de controle de serviços (SCP) e políticas de controle de recursos (RCP).

Alternativas ao elemento **`NotPrincipal`**

Ao gerenciar o controle de acesso em AWS, pode haver cenários em que talvez seja necessário negar explicitamente o acesso de todas as entidades principais a um recurso, exceto uma ou mais entidades principais que você especificar. A AWS recomenda o uso de uma instrução Deny com chaves de contexto de condição global para um controle mais preciso e uma solução de problemas mais fácil. Os exemplos a seguir mostram abordagens alternativas usando operadores de condição, como `StringNotEquals` ou `ArnNotEquals`, para negar acesso a todas as entidades principais, exceto aquelas especificadas no elemento Condition.

Exemplo de cenário usando um perfil do IAM

É possível usar uma política baseada em recursos com uma instrução Deny para impedir que todos os perfis do IAM, exceto aqueles especificados no elemento Condition, acessem ou manipulem seus recursos. Essa abordagem segue o princípio de segurança da AWS em que uma negação explícita sempre tem precedência sobre qualquer declaração de permissão e ajuda a manter o princípio de privilégio mínimo em sua infraestrutura da AWS.

Em vez de usar `NotPrincipal`, recomendamos usar uma instrução Deny com chaves de contexto de condição globais e o operador de condição como `ArnNotEquals` para permitir explicitamente que um perfil do IAM acesse seus recursos. O exemplo a seguir usa aws:PrincipalArn para

permitir explicitamente que o perfil `read-only-role` acesse buckets do Amazon S3 na pasta `Bucket_Account_Audit`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyCrossAuditAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::Bucket_Account_Audit",
        "arn:aws:s3:::Bucket_Account_Audit/*"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:role/read-only-role"
        }
      }
    }
  ]
}
```

Exemplo de cenário usando uma entidade principal de serviço

É possível usar uma instrução Deny para impedir que todas as entidades principais de serviço, exceto aquelas especificados no elemento `Condition`, acessem ou manipulem seus recursos. Essa abordagem é particularmente útil quando você precisa implementar controles de acesso refinados ou estabelecer limites de segurança entre diferentes serviços e aplicações em seu ambiente da AWS.

Em vez de usar `NotPrincipal`, recomendamos usar uma instrução Deny com chaves de contexto de condição globais e o operador de condição `StringNotEquals` para permitir explicitamente que uma entidade principal de serviço acesse seus recursos. O exemplo a seguir usa `aws:PrincipalServiceName` para permitir explicitamente que a entidade principal de serviço AWS CodeBuild acesse buckets do Amazon S3 na pasta `BUCKETNAME`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNotCodeBuildAccess",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME",
        "arn:aws:s3:::BUCKETNAME/*"
      ],
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:PrincipalServiceName": "codebuild.amazonaws.com"
        }
      }
    }
  ]
}
```

Elementos de política JSON do IAM: Action

O elemento `Action` descreve a ação ou ações específicas que serão permitidas ou negadas. As instruções devem incluir um elemento `Action` ou `NotAction`. Cada serviço da AWS tem seu próprio conjunto de ações que descrevem as tarefas que você pode executar com aquele serviço. Por exemplo, a lista de ações do Amazon S3 está disponível em Especificação de permissões em uma política no Guia do usuário do Amazon Simple Storage Service, a lista de ações do Amazon EC2 está disponível em Referência de API do Amazon EC2 e a lista de ações do AWS Identity and Access Management está disponível em Referência de API do IAM. Para encontrar a lista de ações para outros serviços, consulte a documentação de referência de APIs do serviço.

AWSA também fornece informações de referência sobre o serviço no formato JSON para agilizar a automação dos fluxos de trabalho de gerenciamento de políticas. Com as informações de referência do serviço, você pode acessar, em arquivos legíveis por máquina, as ações, recursos e chaves de condições disponíveis nos Serviços da AWS. Para obter mais informações, consulte Simplified AWS service (Serviço da AWS) information for programmatic access na Service Authorization Reference.

Você especifica um valor usando um namespace de serviço, como um prefixo de ação (`iam`, `ec2` `sqs`, `sns`, `s3` etc.) seguido pelo nome da ação para permitir ou negar. O nome deve corresponder a uma ação compatível com o serviço. O prefixo e o nome da ação não diferenciam entre letras maiúsculas e minúsculas. Por exemplo, `iam:ListAccessKeys` é o mesmo que `IAM:listaccesskeys`. Os exemplos a seguir mostram elementos `Action` para diferentes serviços.

Ação do Amazon SQS

```
"Action": "sqs:SendMessage"
```

Ação do Amazon EC

```
"Action": "ec2:StartInstances"
```

Ação do IAM

```
"Action": "iam:ChangePassword"
```

Ação do Amazon S

```
"Action": "s3:GetObject"
```

Você pode especificar vários valores para o elemento `Action`.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Você pode usar curingas de correspondência de vários caracteres (*) e curingas de correspondência de caractere único (?) para conceder acesso a todas as ações que o produto específico da AWS oferece. Por exemplo, o seguinte elemento `Action` se aplica a todas as ações do S3.

```
"Action": "s3:*"
```

Você também pode usar curingas (* ou ?) como parte do nome da ação. Por exemplo, o elemento `Action` a seguir se aplica a todas as ações do IAM que incluem a string `AccessKey`, incluindo `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys` e `UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Alguns serviços permitem que você limite as ações que estão disponíveis. Por exemplo, o Amazon SQS permite que você disponibilize apenas um subconjunto de todas as ações possíveis do Amazon SQS. Neste caso, o curinga * não permite o controle completo da fila; ele permite apenas o subconjunto de ações que você compartilhou. Para obter mais informações, consulte Noções básicas sobre permissões no Guia do desenvolvedor do Amazon Simple Queue Service.

Elementos de política JSON do IAM: NotAction

`NotAction` é um elemento de política avançado que explicitamente corresponde a tudo exceto a lista especificada de ações. O uso de `NotAction` pode resultar em uma política mais curta ao listar apenas algumas ações que não devem corresponder, em vez de incluir uma longa lista de ações para correspondência. As ações especificadas em `NotAction` não são afetadas por `Allow` ou `Deny` em uma instrução de política. Isso, por sua vez, significa que todas as ações ou serviços aplicáveis que não são listados são permitidos se você usar o efeito `Allow`. Além disso, essas ações ou serviços não listados são negados se você usar o efeito `Deny`. Ao usar `NotAction` com o elemento `Resource`, você fornece escopo para a política. Isso é como a AWS determina quais ações ou serviços são aplicáveis. Para obter mais informações, consulte o seguinte exemplo de política.

NotAction com permitir

Você pode usar o elemento `NotAction` em uma instrução com `"Effect": "Allow"` para fornecer acesso a todas as ações em um serviço da AWS, exceto para as ações especificadas em `NotAction`. Você pode usá-lo com o elemento `Resource` para fornecer escopo para a política, limitando as ações permitidas para as ações que podem ser realizadas no recurso especificado.

O exemplo a seguir permite que os usuários acessem todas as ações do Amazon S3 que podem ser executadas em qualquer recurso do S3, exceto a exclusão de um bucket. Essa política também não permite ações em outros serviços, pois outras ações de serviço não são aplicáveis aos recursos do S3.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

Às vezes, você pode querer permitir o acesso a um grande número de ações. O uso do elemento `NotAction` efetivamente reverte a instrução, resultando em uma lista de ações mais curta. Por

exemplo, como há muitos produtos da AWS, você pode criar uma política que permita ao usuário fazer tudo, exceto acessar ações do IAM.

O exemplo a seguir permite aos usuários acessar todas as ações em todos os produtos da AWS, exceto o IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Tenha cuidado ao usar o elemento `NotAction` e `"Effect": "Allow"` na mesma instrução ou em outra instrução dentro de uma política. `NotAction` corresponde a todos os serviços e ações que não são explicitamente listados ou aplicáveis para o recurso especificado e pode resultar em concessão de mais permissões aos usuários do que o desejado.

NotAction com Deny

Você pode usar o elemento `NotAction` em uma instrução com `"Effect": "Deny"` para negar acesso a todos os recursos listados, exceto para as ações especificadas no elemento `NotAction`. Essa combinação não permite os itens listados, mas explicitamente nega as ações não listadas em vez disso. Você ainda deve habilitar as ações que você deseja permitir.

O exemplo condicional a seguir nega acesso a ações que não são do IAM se o usuário não estiver conectado usando a MFA. Se o usuário estiver conectado com MFA, o teste de `"Condition"` falhará e a instrução `"Deny"` final não terá efeito. Observe, no entanto, que isso não concederia ao usuário acesso a qualquer ação, e apenas negaria explicitamente todas as outras ações, exceto as ações do IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
```

```
}
```

Para obter um exemplo de política que nega o acesso a ações fora de regiões específicas, exceto ações de serviços específicos, consulte AWS: nega acesso à AWS com base na região solicitada.

Elementos de política JSON do IAM: Resource

O elemento `Resource` em uma declaração de política do IAM define o objeto ou objetos aos quais a declaração se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`.

Você especifica um recurso usando um nome do recurso da Amazon (ARN). O formato do ARN depende do AWS service (Serviço da AWS) e do recurso específico ao qual você está se referindo. Embora o formato do ARN varie, você sempre usará um ARN para identificar um recurso. Para obter mais informações sobre o formato de ARNs, consulte ARNs do IAM. Para obter informações sobre como especificar um recurso, consulte a documentação do serviço para o qual deseja escrever uma instrução.

> **Note**
>
> Alguns Serviços da AWS não permitem que você especifique ações para recursos individuais. Nesses casos, todas as ações listadas no elemento `Action` ou `NotAction` se aplicam a todos os recursos desse serviço. Quando for esse o caso, você usará o caractere curinga (*) no elemento `Resource`.

O exemplo a seguir se refere a uma fila específica do Amazon SQS.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

O exemplo a seguir faz referência ao usuário do IAM chamado Bob em uma Conta da AWS.

> **Note**
>
> Em um elemento `Resource`, o nome de usuário do IAM diferencia letras maiúsculas de minúsculas.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Uso de caracteres curinga em ARNs de recursos

Você pode usar caracteres curinga (* e ?) dentro dos segmentos individuais de um ARN (as partes separadas por dois pontos) para representar:

- Qualquer combinação de caracteres (*)
- Qualquer caractere único (?)

Você pode usar vários caracteres * ou ? em cada segmento. Se o curinga * for o último caractere de um segmento de ARN de um recurso, ele poderá expandir a busca de correspondência para além dos limites de dois-pontos. Recomendamos que você use curingas (* e ?) dentro dos segmentos de ARN separados por dois pontos.

> **Note**
>
> Você não pode usar um caractere curinga no segmento do serviço que identifica o produto da AWS. Para obter mais informações sobre ARN, consulte Identifique recursos da AWS com nomes do recurso da Amazon (ARNs)

O exemplo a seguir se refere a todos os usuários do IAM cujo caminho é `/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

O exemplo a seguir se refere a todos os itens dentro de um bucket do Amazon S3 específico.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
```

O caractere asterisco (*) pode ser expandido para substituir tudo dentro de um segmento, incluindo caracteres como uma barra (/) que pode parecer um delimitador dentro de um determinado namespace de serviço. Por exemplo, considere o ARN do Amazon S3 a seguir, pois a mesma lógica de expansão de curinga se aplica a todos os serviços.

```
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*/test/*"
```

Os caracteres curinga no ARN se aplicam a todos os objetos a seguir no bucket, não apenas ao primeiro objeto listado.

```
amzn-s3-demo-bucket/1/test/object.jpg
amzn-s3-demo-bucket/1/2/test/object.jpg
amzn-s3-demo-bucket/1/2/test/3/object.jpg
amzn-s3-demo-bucket/1/2/3/test/4/object.jpg
amzn-s3-demo-bucket/1///test///object.jpg
amzn-s3-demo-bucket/1/test/.jpg
amzn-s3-demo-bucket//test/object.jpg
amzn-s3-demo-bucket/1/test/
```

Considere os dois últimos objetos na lista anterior. Um nome de objeto do Amazon S3 pode começar ou terminar com o caractere de barra (/) delimitador convencional. Embora / funcione como um delimitador, não há significado específico quando esse caractere é usado em um ARN de recurso. Ele é tratado da mesma forma que qualquer outro caractere válido. O ARN não corresponderá aos seguintes objetos:

```
amzn-s3-demo-bucket/1-test/object.jpg
amzn-s3-demo-bucket/test/object.jpg
amzn-s3-demo-bucket/1/2/test.jpg
```

Especificação de vários recursos

Você pode especificar vários recursos no elemento `Resource` usando uma matriz de ARNs. O exemplo a seguir se refere a duas tabelas do DynamoDB.

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Uso de variáveis de política em ARNs de recursos

No elemento `Resource`, você pode usar variáveis de política JSON na parte do ARN que identifica o recurso específico, ou seja, na parte final do ARN. Por exemplo, você pode usar a chave `{aws:username}` como parte de um ARN de recurso para indicar que o nome do usuário atual deve ser incluído como parte do nome do recurso. O exemplo a seguir mostra como você pode usar a chave `{aws:username}` em um elemento `Resource`. A política permite o acesso a uma tabela do Amazon DynamoDB que corresponde ao nome do usuário atual.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "dynamodb:*",
        "Resource": "arn:aws:dynamodb:us-east-2:111122223333:table/
${aws:username}"
    }
}
```

Para obter mais informações sobre variáveis de política JSON, consulte Elementos de política do IAM: variáveis e etiquetas.

Elementos de política JSON do IAM: NotResource

`NotResource` é um elemento de política avançado que corresponde explicitamente a cada recurso, exceto aqueles especificados. O uso de `NotResource` pode resultar em uma política mais curta ao listar somente alguns recursos que não devem corresponder, em vez de incluir uma longa lista de recursos para correspondência. Isso é especialmente útil para políticas aplicáveis em um único serviço da AWS.

Por exemplo, imagine que você tenha um grupo chamado `HRPayroll`. Os membros de `HRPayroll` não devem ter permissão para acessar os recursos do Amazon S3, exceto a pasta `Payroll` no bucket `HRBucket`. A seguinte política nega explicitamente o acesso a todos os recursos do Amazon S3, exceto os recursos listados. Observe, no entanto, que essa política não concede ao usuário acesso a qualquer recurso.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
```

```
    ]
  }
}
```

Normalmente, para negar explicitamente o acesso a um recurso, você deve escrever uma política que utilize `"Effect":"Deny"` e que inclua um elemento `Resource` listando cada pasta individualmente. Entretanto, nesse caso, cada vez que você adiciona uma pasta ao `HRBucket`, ou adiciona um recurso ao Amazon S3 que não deve ser acessado, você deve adicionar o nome desses itens à lista no elemento `Resource`. Se você usar um elemento `NotResource` em vez disso, os usuários terão o acesso a novas pastas automaticamente negado, a menos que você adicione os nomes das pastas no elemento `NotResource`.

Ao usar `NotResource`, você deve ter em mente que os recursos especificados neste elemento são os únicos recursos que não são limitados. Isso, por sua vez, limita todos os recursos que seriam aplicáveis à ação. No exemplo acima, a política afeta apenas as ações do Amazon S3 e, portanto, apenas os recursos do Amazon S3. Se o elemento `Action` também incluísse ações do Amazon EC2, a política negaria acesso a nenhum recurso do EC2 não especificados no elemento `NotResource`. Para saber quais ações em um serviço permitem especificar o ARN de um recurso, consulte Ações, recursos e chaves de condição de serviços da AWS.

NotResource com outros elementos

Nunca use os elementos `"Effect": "Allow"`, `"Action": "*"` e `"NotResource": "arn:aws:s3:::HRBucket"` juntos. Essa instrução é muito perigosa porque permite todas as ações na AWS em todos os recursos, exceto no bucket `HRBucket` do S3. Isso permitiria até mesmo que o usuário adicionasse uma política para si mesmo que permitisse o acesso ao `HRBucket`. Não faça isso.

Tenha cuidado ao usar o elemento `NotResource` e `"Effect": "Allow"` na mesma instrução ou em outra instrução em uma política. O `NotResource` permite todos os serviços e recursos que não são explicitamente listados e pode resultar na concessão de mais permissões aos usuários do que o desejado. O uso do elemento `NotResource` e `"Effect": "Deny"` na mesma instrução nega os serviços e recursos que não são explicitamente listados.

Elementos de política JSON do IAM: Condition

O elemento `Condition` (ou bloco `Condition`) permite que você especifique as condições sob as quais uma política está em vigor. O elemento `Condition` é opcional. No elemento `Condition`, crie

expressões em que você usa operadores de condição (equal, less than, entre outros) para fazer a correspondência de chaves e valores de contexto na política com os valores e chaves no contexto da solicitação. Para saber mais sobre o contexto da solicitação, consulte Componentes de uma solicitação.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

A chave de contexto especificada em uma política de condição pode ser uma chave de contexto de condição global ou uma chave de contexto específica do serviço. As chaves de contexto de condição globais têm o prefixo `aws:`. As chaves de contexto específicas de serviços têm o prefixo do serviço. Por exemplo, o Amazon EC2 permite que você escreva uma condição usando a chave de contexto `ec2:InstanceType`, que é exclusiva para esse serviço. Para visualizar as chaves de contexto do IAM específicas do serviço com o prefixo `iam:`, consulte Chaves de contexto de condição do IAM e do AWS STS.

Os nomes de chave de contexto não diferenciam maiúsculas de minúsculas. Por exemplo, incluir a chave de contexto `aws:SourceIP` é equivalente a testar o elemento `AWS:SourceIp`. A diferenciação de maiúsculas e minúsculas nos valores da chave de contexto depende do operador de condição que você usa. Por exemplo, a condição a seguir inclui o operador `StringEquals` para garantir que apenas solicitações feitas por `john` sejam aceitas. Os usuários chamados `John` não têm permissão de acesso.

```
"Condition" : { "StringEquals" : { "aws:username" : "john" }}
```

A seguinte condição usa o operador `StringEqualsIgnoreCase` para corresponder usuários chamados `john` ou `John`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "john" }}
```

Algumas chaves de contexto são compatíveis com pares de chave-valor que permitem especificar parte do nome da chave. Os exemplos incluem a chave de contexto `aws:RequestTag/tag-key`, a `kms:EncryptionContext:encryption_context_key` do AWS KMS e a chave de contexto `ResourceTag/tag-key` compatível com múltiplos serviços.

- Se você usar a chave de contexto `ResourceTag/tag-key` para um serviço, como o Amazon EC2, será necessário especificar um nome de chave para `tag-key`.
- Os nomes de chave não diferenciam maiúsculas de minúsculas. Isso significa que, se você especificar `"aws:ResourceTag/TagKey1": "Value1"` no elemento de condição da política, a

condição corresponderá a uma chave de tag de recurso chamada `TagKey1` ou `tagkey1`, mas não ambas.

- AWSOs serviços da compatíveis com esses atributos podem permitir que você crie vários nomes de chave que diferem apenas por maiúsculas e minúsculas. Por exemplo, é possível etiquetar uma instância do Amazon EC2 com `ec2=test1` e `EC2=test2`. Quando você usa uma condição, como `"aws:ResourceTag/EC2": "test1"`, para permitir o acesso a esse recurso, o nome da chave corresponde a ambas as tags, mas apenas um valor é correspondente. Isso pode resultar em falhas de condição inesperadas.

> **Important**
>
> Como prática recomendada, certifique-se de que os membros de sua conta sigam uma convenção de nomenclatura consistente ao nomear atributos de par de chave-valor. Os exemplos incluem tags ou contextos de criptografia do AWS KMS. Você pode aplicar isso usando a chave de contexto `aws:TagKeys` para a marcação, ou a `kms:EncryptionContextKeys` para o contexto de criptografia do AWS KMS.

- Para obter uma lista de todos os operadores de condição e uma descrição de como eles funcionam, consulte Operadores de condição.
- A menos que especificado de outra forma, todas as chaves de contexto podem ter múltiplos valores. Para obter uma descrição de como lidar com chaves de contexto com múltiplos valores, consulte Operadores de conjunto para chaves de contexto de vários valores.
- Para obter uma lista de todas as chaves de contexto disponíveis globalmente, consulte AWSChaves de contexto de condições globais da.
- Para as chaves de contexto de condição definidas por cada serviço, consulte Ações, recursos e chaves de condição dos serviços da AWS.

O contexto da solicitação

Quando um principal faz uma solicitação à AWS, a AWS reúne as informações da solicitação em um contexto de solicitação. O contexto da solicitação inclui informações sobre a entidade principal, os recursos, as ações e outras propriedades ambientais. A avaliação da política compara as propriedades da política com as propriedades enviadas na solicitação para avaliar e autorizar ações que você pode realizar na AWS.

É possível usar o elemento `Condition` de uma política JSON para testar chaves de contexto específicas em relação ao contexto da solicitação. Por exemplo, é possível criar uma política que use a chave de contexto aws:CurrentTime para permitir que um usuário execute ações somente durante um intervalo de datas.

O exemplo a seguir mostra uma representação do contexto da solicitação quando Martha Rivera envia uma solicitação para desativar o seu dispositivo de MFA.

```
Principal: AROA123456789EXAMPLE
Action: iam:DeactivateMFADevice
Resource: arn:aws:iam::user/martha
Context:
  - aws:UserId=AROA123456789EXAMPLE:martha
  - aws:PrincipalAccount=1123456789012
  - aws:PrincipalOrgId=o-example
  - aws:PrincipalARN=arn:aws:iam::1123456789012:assumed-role/TestAR
  - aws:MultiFactorAuthPresent=true
  - aws:MultiFactorAuthAge=2800
  - aws:CurrentTime=...
  - aws:EpochTime=...
  - aws:SourceIp=...
```

O contexto da solicitação é comparado a uma política que permite que os usuários removam seu próprio dispositivo de autenticação multifator (MFA), mas somente se fizeram login usando MFA na última hora (3.600 segundos).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
```

```
}
```

Neste exemplo, a política corresponde ao contexto da solicitação: a ação é a mesma, o recurso corresponde ao caractere curinga "*" e o valor para `aws:MultiFactorAuthAge` é 2800, que é menor que 3600, portanto, a política permite essa solicitação de autorização.

A AWS avalia cada chave de contexto na política e retorna um valor verdadeiro ou falso. Uma chave de contexto que não está na solicitação é considerada uma não correspondência.

O contexto da solicitação pode retornar os seguintes valores:

- True (verdadeiro): se o solicitante tiver feito login usando MFA na última hora ou menos, a condição retornará true.
- False (falso): se o solicitante tiver feito login usando MFA há mais de uma hora, a condição retorna false.
  - Not present (não presente): se o solicitante tiver feito uma solicitação usando suas chaves de acesso do usuário do IAM na AWS CLI ou na API da AWS, a chave não estará presente. Nesse caso, a chave não estará presente e não será correspondente.

> **Note**
>
> Em alguns casos, quando o valor da chave de condição não está presente, a condição ainda pode retornar verdadeira. Por exemplo, se você adicionar o qualificador `ForAllValues`, a solicitação retornará verdadeira se a chave de contexto não estiver na solicitação. Para evitar que chaves de contexto ausentes ou chaves de contexto com valores vazios sejam avaliadas como verdadeiras, você pode incluir o operador de condição Null em sua política com um valor `false` para verificar se a chave de contexto existe e se seu valor não for nulo.

O bloco de condição

O exemplo a seguir mostra o formato básico de um elemento `Condition`:

```
"Condition": {"StringLike": {"s3:prefix": ["jane/*"]}}
```

Um valor da solicitação é representado por uma chave de contexto, neste caso, `s3:prefix`. O valor da chave de contexto é comparado a um valor especificado como um valor literal, como `jane/*`.

O tipo de comparação a fazer é especificado pelo operador de condição (aqui `StringLike`). Você pode criar condições que comparam strings, datas, números etc., usando comparações boolianas típicas, comoigual a, maior que e menor que. Ao usar operadores de string ou operadores de ARN, você também pode usar uma variável de política no valor da chave de contexto. O exemplo a seguir inclui a variável `aws:username`.

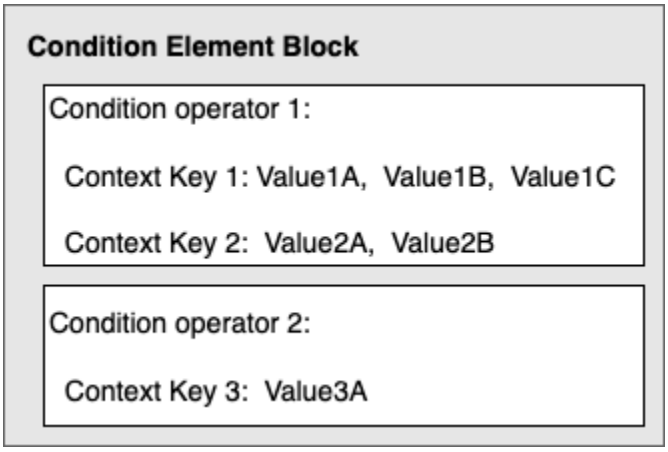
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

Em algumas circunstâncias, as chaves de contexto podem conter múltiplos valores. Por exemplo, uma solicitação ao Amazon DynamoDB pode solicitar o retorno ou a atualização de vários atributos de uma tabela. Uma política de acesso às tabelas do DynamoDB pode incluir a chave de contexto `dynamodb:Attributes`, que contém todos os atributos listados na solicitação. Você pode testar os múltiplos atributos na solicitação em relação a uma lista de atributos permitidos em uma política usando operadores de conjunto no elemento `Condition`. Para obter mais informações, consulte Operadores de conjunto para chaves de contexto de vários valores.

Quando a política é avaliada durante uma solicitação, a AWS substitui a chave pelo valor correspondente da solicitação. (Neste exemplo, a AWS usaria a data e a hora da solicitação.) A condição é avaliada para retornar verdadeiro ou falso, o que, então, é considerado ao avaliar se a política como um todo permite ou nega a solicitação.

Vários valores em uma condição

Um elemento `Condition` pode conter múltiplos operadores de condição, e cada operador de condição pode conter múltiplos pares de chave-valor de contexto. A figura a seguir ilustra isso.



Para obter mais informações, consulte Operadores de conjunto para chaves de contexto de vários valores.

Elementos de política JSON do IAM: operadores de condição

Use operadores de condição no elemento `Condition` para corresponder a chave de condição e o valor na política aos valores no contexto da solicitação. Para obter mais informações sobre o elemento `Condition`, consulte Elementos de política JSON do IAM: Condition.

O operador de condição que você pode usar em uma política depende da chave de condição escolhida. É possível escolher uma chave de condição global ou uma chave de condição específica do serviço. Para saber qual operador de condição pode ser usado para uma chave de condição global, consulte AWSChaves de contexto de condições globais da. Para saber qual operador de condição você pode usar para uma chave de condição específica de serviço, consulte Ações, recursos e chaves de condição para produtos da AWS e escolha o serviço que deseja visualizar.

> **Important**
>
> Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão e a condição será falsa. Se a condição da política exigir que a chave não seja correspondida, como `StringNotLike` ou `ArnNotLike`, e a chave certa não estiver presente, a condição será verdadeira. Esta lógica se aplica a todos os operadores de condição, exceto ...IfExists e Null check. Esses operadores testam se a chave está presente (existe) no contexto da solicitação.

Os operadores de condição podem ser agrupados nas seguintes categorias:

- String
- Numérico
- Data e hora
- Booliano
- Binário
- Endereço IP
- Nome de recurso da Amazon (ARN) (disponível apenas para alguns serviços.)
- ... IfExists (verifica se o valor da chave existe como parte de outra verificação)
- Verificação de Null (verifica se o valor da chave existe como uma verificação independente)

Operadores de condição de strings

Operadores de condição de string permitem que você construa elementos `Condition` que restringem o acesso com base na comparação de uma chave a um valor de string.

- Variáveis de política: compatíveis
- Curingas: compatíveis

Operador de condição	Descrição
`StringEquals`	Correspondência exata, distinção entre letras maiúsculas e minúsculas
`StringNotEquals`	Correspondência negativa
`StringEqualsIgnoreCase`	Correspondência exata, sem distinção entre letras maiúsculas e minúsculas
`StringNotEqualsIgnoreCase`	Correspondência negativa, sem distinção entre letras maiúsculas e minúsculas
`StringLike`	Correspondência com distinção entre letras maiúsculas e minúsculas. Os valores podem incluir uma correspondência com vários caracteres curinga (*) e uma correspondência com um único caractere curinga (?) em qualquer ponto da string. Você deve especificar curingas para obter correspondências parciais de strings. **Note** Se uma chave contiver vários valores, `StringLike` poderá ser qualificado com os operadores de conjunto `ForAllValues:StringLike` e `ForAnyValue:StringLike`. Para obter mais informações, consulte Operadores de conjunto para chaves de contexto de vários valores.

Operador de condição	Descrição
`StringNotLike`	Correspondência negativa com distinção entre letras maiúsculas e minúsculas. Os valores podem incluir uma correspondência com vários caracteres curinga (*) ou uma correspondência com um único caractere curinga (?) em qualquer ponto da string.

Example operador de condição string

Por exemplo, a declaração a seguir contém um elemento `Condition` que usa a chave `aws:PrincipalTag` para especificar que a entidade principal que está fazendo a solicitação deve ser marcada com a categoria de trabalho `iamuser-admin`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalTag/job-category": "iamuser-admin"
            }
        }
    }
}
```

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão. Nesse exemplo, a chave `aws:PrincipalTag/job-category` estará presente no contexto da solicitação se a entidade de segurança estiver usando um usuário do IAM com etiquetas anexadas. Ela também será incluída para um principal usando uma função do IAM com tags anexadas ou tags de sessão. Se um usuário sem a tag tentar visualizar ou editar uma chave de acesso, a condição retornará `false` e a solicitação será implicitamente negada por essa declaração.

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	`aws:PrincipalTag/job-category: - iamuser-admin`	correspondência
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	`aws:PrincipalTag/job-category: - dev-ops`	Nenhuma correspondência
`"StringEquals": { "aws:PrincipalTag/job-category": "iamuser-admin" }`	Nenhuma `aws:PrincipalTag/job-category` no contexto da solicitação.	Nenhuma correspondência

Example usar uma variável de política com um operador de condição string

O exemplo a seguir usa o operador de condição `StringLike` para realizar correspondência de string com uma variável de política para criar uma política que permite a um usuário do IAM usar o console do Amazon S3 para gerenciar seu próprio “diretório base” em um bucket do Amazon S3. A política permite as ações especificadas em um bucket do S3, desde que o `s3:prefix` corresponda a qualquer um dos padrões especificados.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "",
            "home/",
            "home/${aws:username}/"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}",
        "arn:aws:s3:::amzn-s3-demo-bucket/home/${aws:username}/*"
      ]
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política para diferentes usuários com base no valor aws:username no contexto da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringLike": { "s3:prefix": [ "home/",`	`aws:username: - martha_rivera`	`"StringLike": { "s3:prefix": [ "home/",`

Condição da política	Contexto da solicitação	Resultado

`"home/${aws:username}/"`
`]`
`}`
		`"home/martha_rivera /"`
		`]`
		`}`

`"StringLike": {`
`"s3:prefix": [`
`"home/",`
`"home/${aws:username}/"`
`]`
`}`
	`aws:username:`
	`- nikki_wolf`
		`"StringLike": {`
		`"s3:prefix": [`
		`"home/",`
		`"home/nikki_wolf /"`
		`]`
		`}`

`"StringLike": {`
`"s3:prefix": [`
`"home/",`
`"home/${aws:username}/"`
`]`
`}`
	Nenhum `aws:username` no contexto da solicitação.	Nenhuma correspondência

Para obter um exemplo de uma política que mostra como usar o elemento `Condition` para restringir o acesso a recursos com base em um ID de aplicação e um ID de usuário para a federação de OIDC, consulte Amazon S3: permite que usuários do Amazon Cognito acessem objetos em seus buckets.

Operadores de condição string com vários valores

Se uma chave da solicitação contiver vários valores, os operadores string poderão ser qualificados com os operadores de conjunto `ForAllValues` e `ForAnyValue`. Para obter mais informações sobre a lógica de avaliação de várias chaves ou valores de contexto, consulte Operadores de conjunto para chaves de contexto de vários valores.

Operador de condição	Descrição

`ForAllValues:StringEquals`
 	
`ForAllValues:StringEqualsIgnoreCase`
	Todos os valores da chave de condição da solicitação devem corresponder a pelo menos um valor da política.

`ForAnyValue:StringEquals`
 	
`ForAnyValue:StringEqualsIgnoreCase`
	Pelo menos um valor de chave de condição da solicitação deve corresponder a um valor da política.

`ForAllValues:StringNotEquals`
 	
`ForAllValues:StringNotEqualsIgnoreCase`
	Correspondência negada.
	
	Nenhum valor da chave de contexto da solicitação pode corresponder a algum valor de chave de contexto da política.

`ForAnyValue:StringNotEquals`
 	
`ForAnyValue:StringNotEqualsIgnoreCase`
	Correspondência negada.
	
	Pelo menos um valor de chave de contexto da solicitação NÃO deve corresponder a nenhum valor da chave de contexto da política.

`ForAllValues:StringLike`	Todos os valores da chave de condição da solicitação devem corresponder a pelo menos um valor da política.
`ForAnyValue:StringLike`	Pelo menos um valor de chave de condição da solicitação deve corresponder a um valor da política.
`ForAllValues:StringNotLike`	Correspondência negada. Nenhum valor da chave de contexto da solicitação pode corresponder a algum valor de chave de contexto da política.
`ForAnyValue:StringNotLike`	Correspondência negada.

Operador de condição	Descrição
	Pelo menos um valor de chave de contexto da solicitação NÃO deve corresponder a nenhum valor da chave de contexto da política.

Example usando **`ForAnyValue`** com um operador de condição string

Este exemplo mostra como você pode criar uma política baseada em identidade que permita usar a ação `CreateTags` do Amazon EC2 para anexar tags a uma instância. Quando você usa `StringEqualsIgnoreCase`, somente pode anexar uma tag se ela contiver a chave `environment` com os valores `preprod` ou `storage`. Quando você acrescenta `IgnoreCase` ao operador, permite que qualquer capitalização de valor de tag existente, como `preprod`, `Preprod` e `PreProd`, seja resolvida como verdadeira.

Quando você adiciona o modificador `ForAnyValue` com a chave de condição aws:TagKeys, pelo menos um valor de chave de tag da solicitação deve corresponder ao valor `environment`. A comparação `ForAnyValue` diferencia maiúsculas de minúsculas, o que impede que os usuários usem maiúsculas e minúsculas incorretamente na chave da tag, por exemplo, usar `Environment` em vez de `environment`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:*:*:instance/*",
    "Condition": {
      "StringEqualsIgnoreCase": {
        "aws:RequestTag/environment": [
          "preprod",
          "storage"
        ]
      },
      "ForAnyValue:StringEquals": {
        "aws:TagKeys": "environment"
      }
    }
```

```
  }
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment aws:RequestTag/environment: - preprod`	correspondência
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment - costcenter aws:RequestTag/environment: - PreProd`	correspondência
`"StringEqualsIgnoreCase": {`	`aws:TagKeys: - Environment`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado

<pre>"aws:RequestTag/environment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEquals": {
"aws:TagKeys":
"environment"
}</pre>
	<pre>aws:RequestTag/
	Environment:
	- preprod</pre>	

<pre>"StringEqualsIgnoreCase": {
"aws:RequestTag/environment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEquals": {
"aws:TagKeys":
"environment"
}</pre>
	<pre>aws:TagKeys:
	- costcenter
	aws:RequestTag/environment:
	- preprod</pre>
		Nenhuma correspondência

<pre>"StringEqualsIgnoreCase": {
"aws:RequestTag/environment": [
"preprod",
"storage"
]
},
"ForAnyValue:StringEquals": {
"aws:TagKeys":
"environment"
}</pre>
	Nenhum `aws:TagKeys` no contexto da solicitação.
	<pre>aws:RequestTag/environment:
	- storage</pre>
		Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	`aws:TagKeys: - environment` Nenhum `aws:RequestTag/environment` no contexto da solicitação.	Nenhuma correspondência
`"StringEqualsIgnoreCase": { "aws:RequestTag/environment": [ "preprod", "storage" ] }, "ForAnyValue:StringEquals": { "aws:TagKeys": "environment" }`	Nenhum `aws:TagKeys` no contexto da solicitação. Nenhum `aws:RequestTag/environment` no contexto da solicitação.	Nenhuma correspondência

Correspondência de curinga

Os operadores de condição de string realizam uma correspondência sem padrões que não impõe um formato predefinido. Os operadores de condição ARN e Date são um subconjunto de operadores de string que impõem uma estrutura no valor da chave de condição.

Recomendamos usar operadores de condição que correspondam aos valores com os quais você está comparando as chaves. Por exemplo, você deve usar the section called “Operadores de condição de strings” ao comparar chaves com valores de string. Da mesma forma, você deve usar

the section called “Operadores de condição de nome do recurso da Amazon (ARN)” ao comparar chaves com valores de ARN.

Example

Este exemplo mostra como é possível criar uma barreira em torno dos recursos de sua organização. A condição dessa política nega acesso às ações do Amazon S3, a menos que o recurso sendo acessado esteja em um conjunto específico de unidades organizacionais (UOs) em AWS Organizations. Um caminho do AWS Organizations é uma representação textual da estrutura da entidade da organização.

A condição exige que `aws:ResourceOrgPaths` contenha algum dos caminhos de UO listados. Como `aws:ResourceOrgPaths` é uma condição com vários valores, a política usa o operador `ForAllValues:StringNotLike` para comparar os valores de `aws:ResourceOrgPaths` com a lista de UOs da política.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyS3AccessOutsideMyBoundary",
      "Effect": "Deny",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:ResourceOrgPaths": [
            "o-acorg/r-acroot/ou-acroot-mediaou/",
            "o-acorg/r-acroot/ou-acroot-sportsou/*"
          ]
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	`aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-sportsou/costcenter/`	correspondência
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	`aws:ResourceOrgPaths: - o-acorg/r-acroot/ou-acroot-mediaou/costcenter/`	Nenhuma correspondência
`"ForAllValues:StringNotLike": { "aws:ResourceOrgPaths": [ "o-acorg/r-acroot/ou-acroot-mediaou/", "o-acorg/r-acroot/ou-acroot-sportsou/*" ] }`	Nenhum `aws:ResourceOrgPaths:` na solicitação.	No match (Nenhuma correspondência)

Operadores de condição numéricos

Operadores de condição numéricos permitem que você construa elementos `Condition` que restringem o acesso com base na comparação de uma chave a um número inteiro ou valor decimal.

- Variáveis de política: não compatíveis
- Curingas: não compatíveis

Operador de condição	Descrição
`NumericEquals`	Matching
`NumericNotEquals`	Correspondência negativa
`NumericLessThan`	Correspondência "menor que"
`NumericLessThanEquals`	Correspondência "menor ou igual a"
`NumericGreaterThan`	Correspondência "maior que"
`NumericGreaterThanEquals`	Correspondência "maior ou igual a"

Por exemplo, a seguinte instrução contém um elemento `Condition` que usa o operador de condição `NumericLessThanEquals` com a chave `s3:max-keys` para especificar que o solicitante pode listar até 10 objetos no `amzn-s3-demo-bucket` por vez.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
```

```
}
```

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão. Neste exemplo, a chave `s3:max-keys` está sempre presente na solicitação ao realizar a operação `ListBucket`. Se essa política permitisse todas as operações do Amazon S3, somente as operações que incluíssem a chave de contexto `max-keys` com um valor inferior ou igual a 10 seriam permitidas.

Operadores de condição de data

Operadores de condição de data permitem que você construa elementos `Condition` que restringem o acesso com base na comparação de uma chave a um valor de data/hora. Você pode usar esses operadores de condição com a chave `aws:CurrentTime` ou a chave `aws:EpochTime`. Você deve especificar os valores de data e hora com uma das implementações W3C dos formatos de hora ISO 8601 ou em data e hora epoch (UNIX).

- Variáveis de política: não compatíveis
- Curingas: não compatíveis

Operador de condição	Descrição
`DateEquals`	Correspondência de uma data específica
`DateNotEquals`	Correspondência negativa
`DateLessThan`	Correspondência antes de uma data e hora específicas
`DateLessThanEquals`	Correspondência antes ou em uma data e hora específicas
`DateGreaterThan`	Correspondência após uma data e hora específicas
`DateGreaterThanEqu als`	Correspondência após ou em uma data e hora específicas

Por exemplo, a instrução a seguir contém um elemento `Condition` que usa o operador de condição `DateGreaterThan` com a chave `aws:TokenIssueTime`. Esta condição especifica que as credenciais de segurança temporárias usadas para fazer a solicitação foram emitidas em 2020. Esta

política pode ser atualizada de forma programática todos os dias para garantir que os membros da conta usem credenciais novas.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "DateGreaterThan": {
                "aws:TokenIssueTime": "2020-01-01T00:00:01Z"
            }
        }
    }
}
```

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão. A chave `aws:TokenIssueTime` está presente no contexto da solicitação somente quando o principal usar as credenciais temporárias para realizar a solicitação. A chave não está presente em solicitações da AWS CLI, da API da AWS ou do AWS SDK que são feitas usando chaves de acesso. Neste exemplo, se um usuário do IAM tentar visualizar ou editar uma chave de acesso, a solicitação será negada.

Operadores de condição booliana

Condições boolianas permitem que você crie elementos `Condition` que restrinjam o acesso com base na comparação de uma chave com `true` ou `false`.

Se uma chave contiver vários valores, os operadores boolianos podem ser qualificados com os operadores de conjunto `ForAllValues` e `ForAnyValue`. Para obter mais informações sobre a lógica de avaliação de várias chaves ou valores de contexto, consulte Operadores de conjunto para chaves de contexto de vários valores.

- Variáveis de política: compatíveis
- Curingas: não compatíveis

Operador de condição	Descrição
`Bool`	Correspondência booliana
`ForAllValues:Bool`	Use com o tipo de dados de matriz booliana. Todos os valores boolianos nos valores de chave de contexto devem corresponder aos valores boolianos da política. Para evitar que operadores `ForAllValues` avaliem chaves de contexto ausentes ou com valores vazios como Permitidas, você pode incluir o operador de condição Null.
`ForAnyValue:Bool`	Use com o tipo de dados de matriz booliana. Pelo menos um valor booliano nos valores de chave de contexto deve corresponder aos valores booleanos da política.

Example operador de condição booliano

A política baseada em identidades a seguir usará o operador de condição `Bool` com a chave `aws:SecureTransport` para negar a replicação de objetos e tags de objetos no bucket de destino e seu conteúdo se a solicitação não for por SSL.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
```

```
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"Bool": { "aws:SecureTranspo rt": "false" }`	`aws:SecureTransport: - false`	correspondência
`"Bool": { "aws:SecureTranspo rt": "false" }`	`aws:SecureTransport: - true`	Nenhuma correspondência
`"Bool": { "aws:SecureTranspo rt": "false" }`	Nenhum `aws:Secur eTransport` no contexto da solicitação.	Nenhuma correspondência

Operadores de condição binários

O operador de condição `BinaryEquals` permite que você crie elementos `Condition` que testam valores de chave no formato binário. Ele compara o valor da chave especificada byte por byte à

uma representação base-64 codificada do valor binário na política. Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão.

- Variáveis de política: não compatíveis
- Curingas: não compatíveis

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Condição da política	Contexto da solicitação	Resultado
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	`key: - QmluYXJ5VmFsdWVJbk Jhc2U2NA==`	correspondência
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	`key: - ASIAIOSFODNN7EXAMP LE`	Nenhuma correspondência
`"BinaryEquals": { "key" : "QmluYXJ5 VmFsdWVJbkJhc2U2NA==" }`	Nenhum `key` no contexto da solicitação.	Nenhuma correspondência

Operadores de condição de endereço IP

Operadores de condição de endereço IP permitem que você construa elementos `Condition` que restringem o acesso com base na comparação de uma chave a um endereço IPv4 ou IPv6 ou a intervalo de endereços IP. Você pode usá-los com a chave `aws:SourceIp`. O valor deve ser no formato CIDR padrão (por exemplo, 203.0.113.0/24 ou 2001: DB8:1234:5678::/64). Se você

especificar um endereço IP, sem o prefixo de roteamento associado, o IAM usará o valor do prefixo padrão `/32`.

Alguns serviços da AWS oferecem suporte a IPv6, usando :: para representar um intervalo de 0s. Para saber se um serviço oferece suporte a IPv6, consulte a documentação do serviço.

- Variáveis de política: não compatíveis
- Curingas: não compatíveis

Operador de condição	Descrição
`IpAddress`	O endereço IP ou intervalo especificado
`NotIpAddress`	Todos os endereços IP, exceto o endereço IP ou intervalo especific ado

Example Operador de condição endereço IP

A instrução a seguir usa o operador de condição `IpAddress` com a chave `aws:SourceIp` para especificar que a solicitação deve ser originada no intervalo de IP 203.0.113.0 a 203.0.113.255.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::111122223333:user/*",
        "Condition": {
            "IpAddress": {
                "aws:SourceIp": "203.0.113.0/24"
            }
        }
    }
}
```

A chave de condição `aws:SourceIp` resulta no endereço IP onde a solicitação foi gerada. Se as solicitações for proveniente de uma instância do Amazon EC2, o `aws:SourceIp` será avaliado para o endereço IP público da instância.

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão. A chave `aws:SourceIp` está sempre presente no contexto da solicitação, exceto quando o solicitante usar um VPC endpoint para fazer a solicitação. Nesse caso, a condição retornará `false` e a solicitação será negada implicitamente por essa declaração.

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 203.0.113.1`	correspondência
`"IpAddress": { "aws:SourceIp": "203.0.113.0/24" }`	`aws:SourceIp: - 198.51.100.1`	Nenhuma correspondência

O exemplo a seguir mostra como combinar endereços IPv4 e IPv6 para cobrir todos os endereços IP válidos da sua organização. Recomendamos atualizar seus intervalos de endereço IPv6, além dos intervalos IPv4 que você já possui, nas políticas de sua organização para garantir que as políticas continuarão a funcionar à medida que você fizer a transição para o IPv6.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
```

```
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

A chave de condição `aws:SourceIp` só funcionará em uma política JSON se você estiver chamando a API testada diretamente como um usuário. Se você usar um serviço para chamar o serviço de destino em seu nome, o serviço de destino vê o endereço IP do serviço de chamada, em vez do endereço IP do usuário-fonte. Isso pode acontecer, por exemplo, se você usar o AWS CloudFormation para chamar o Amazon EC2 para criar instâncias para você. Atualmente, não há como passar o endereço IP de origem através de um serviço de chamada ao serviço de destino para avaliação em uma política JSON. Para esses tipos de chamadas de serviço de API, não use a chave de condição `aws:SourceIp`.

Operadores de condição de nome do recurso da Amazon (ARN)

Operadores de condição do nome de recurso da Amazon (ARN) permitem que você construa elementos `Condition` que restringem o acesso com base na comparação de uma chave a um ARN. O ARN é considerado uma string.

- Variáveis de política: compatíveis
- Curingas: compatíveis

Operador de condição	Descrição
`ArnEquals` , `ArnLike`	Correspondência do ARN com distinção entre letras maiúsculas e minúsculas. Cada um dos seis componentes do ARN delimitados por dois pontos é verificado separadamente e cada um pode incluir de múltiplos caracteres curingas (*) ou um único caractere curinga (?). Os operadores de condição `ArnEquals` e `ArnLike` têm comportam ento semelhante.

Operador de condição	Descrição
`ArnNotEquals` , `ArnNotLike`	Correspondência negativa para ARN. Os operadores de condição `ArnNotEquals` e `ArnNotLike` têm comportamento semelhante.

Example Operador de condição ARN

O exemplo de política baseada em recurso a seguir mostra uma política anexada a uma fila do Amazon SQS para a qual você deseja enviar mensagens do SNS. Ela fornece ao Amazon SNS permissão para enviar mensagens para a fila (ou as filas) de sua escolha, mas apenas se o serviço estiver enviando as mensagens em nome de um determinado tópico (ou tópicos) do Amazon SNS. Você especifica a fila no campo `Resource` e o tópico do Amazon SNS como o valor para a chave `SourceArn`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {
            "Service": "sns.amazonaws.com"
        },
        "Action": "SQS:SendMessage",
        "Resource": "arn:aws:sqs:us-east-1:123456789012:QUEUE-ID",
        "Condition": {
            "ArnEquals": {
                "aws:SourceArn": "arn:aws:sns:us-east-1:123456789012:TOPIC-ID"
            }
        }
    }
}
```

A chave `aws:SourceArn` estará presente no contexto da solicitação somente se um recurso acionar um serviço para chamar outro serviço em nome do proprietário do recurso. Se um usuário do IAM tentar realizar essa operação diretamente, a condição retornará `false` e a solicitação será negada implicitamente por esta instrução.

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:123456789012:TOPIC-ID`	correspondência
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	`aws:SourceArn: - arn:aws:sns:us-west-2:777788889999:TOPIC-ID`	Nenhuma correspondência
`"ArnEquals": { "aws:SourceArn": "arn:aws:sns:us-west-2:123456789012:TOPIC-ID" }`	Nenhum `aws:SourceArn` no contexto da solicitação.	Nenhuma correspondência

Operadores de condição ARN com vários valores

Se uma chave da solicitação contiver vários valores, os operadores ARN podem ser qualificados com os operadores de conjunto `ForAllValues` e `ForAnyValue`. Para obter mais informações sobre a lógica de avaliação de várias chaves ou valores de contexto, consulte Operadores de conjunto para chaves de contexto de vários valores.

Operador de condição	Descrição

`ForAllValues:ArnEquals`
 	
`ForAllValues:ArnLike`
	Todos os ARNs no contexto da solicitação devem corresponder a pelo menos um padrão de ARN da política.

`ForAnyValue:ArnEquals`
 	
`ForAnyValue:ArnLike`
	Pelo menos um ARN no contexto da solicitaç ão deve corresponder a um padrão de ARN da política.

`ForAllValues:ArnNotEquals`
 	
`ForAllValues:ArnNotLike`
	Correspondência negada.
	
	Nenhum ARN no contexto da solicitação pode corresponder a algum padrão de ARN de string da política.

`ForAnyValue:ArnNotEquals`
 	
`ForAnyValue:ArnNotLike`
	Correspondência negada.
	
	Pelo menos um ARN no contexto da solicitação NÃO deve corresponder a nenhum padrão de ARN da política.

Example usando **`ForAllValues`** com um operador de condição ARN

O exemplo a seguir usa `ForAllValues:ArnLike` para criar ou atualizar uma origem de entrega lógica para os logs do Amazon CloudWatch Logs. O bloco de condição inclui a chave de condição `logs:LogGeneratingResourceArns` para filtrar o log que gera os ARNs passados na solicitação. Usando esse operador de condição, todos os ARNs da solicitação devem corresponder a pelo menos um ARN da política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
```

```
            "Action": "logs:PutDeliverySource",
            "Resource": "arn:aws:logs:us-east-1:123456789012:delivery-source:*",
            "Condition": {
                "ForAllValues:ArnLike": {
                    "logs:LogGeneratingResourceArns": [
                        "arn:aws:cloudfront::123456789012:distribution/*",
                        "arn:aws:cloudfront::123456789012:distribution/support*"
                    ]
                }
            }
        }
    ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter`	correspondência
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*",`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/support2025`	correspondência

Condição da política	Contexto da solicitação	Resultado
`"arn:aws::cloudfront:123456789012:distribution/support*" ] }`		
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:123456789012:distribution/costcenter - arn:aws::cloudfront:123456789012:distribution/admin`	correspondência
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	`logs:LogGeneratingResourceArns: - arn:aws::cloudfront:777788889999:distribution/costcenter`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:ArnLike": { "logs:LogGeneratingResourceArns": [ "arn:aws::cloudfront:123456789012:distribution/*", "arn:aws::cloudfront:123456789012:distribution/support*" ] }`	Nenhum `logs:LogGeneratingResourceArns` no contexto da solicitação.	correspondência

O qualificador `ForAllValues` retornará true se não houver chaves de contexto da solicitação ou se o valor da chave de contexto for resolvido para um conjunto de dados nulo, como uma string vazia. Para evitar que chaves de contexto ausentes ou chaves de contexto com valores vazios sejam avaliadas como verdadeiras, você pode incluir o operador de condição Null em sua política com um valor `false` para verificar se a chave de contexto existe e se seu valor não for nulo.

Operadores de condição ...IfExists

Você pode adicionar `IfExists` ao final de qualquer nome de operador de condição, exceto a condição `Null`, por exemplo, `StringLikeIfExists`. Isso é feito para dizer "Se a chave de condição estiver presente no contexto da solicitação, processar a chave conforme especificado na política. Se a chave não estiver presente, avalie o elemento da condição como verdadeiro." Outros elementos de condição na instrução ainda podem resultar em um nonmatch, mas não em uma chave ausente quando marcada com `...IfExists`. Se você estiver usando um elemento `"Effect": "Deny"` com um operador de condição negada como `StringNotEqualsIfExists`, a solicitação ainda será negada mesmo se a chave de condição estiver ausente.

Exemplo usando **`IfExists`**

Muitas chaves de condição descrevem informações sobre determinado tipo de recurso e existem apenas ao acessar aquele tipo de recurso. Essas chaves de condição não estão presentes em outros tipos de recursos. Isso não causa problemas quando a declaração de política se aplica a apenas um tipo de recurso. No entanto, há casos em que uma única instrução pode se aplicar a vários tipos de recursos, tal como quando a declaração de política se refere a ações de múltiplos

serviços ou quando dada ação de um serviço acessa diversos tipos de recursos dentro do mesmo serviço. Em tais casos, a inclusão de uma chave de condição que se aplica a apenas um dos recursos na declaração de política pode fazer com que o elemento `Condition` na declaração de política falhe de forma que o `"Effect"` da instrução não se aplica.

Por exemplo, considere o exemplo de política:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

O objetivo da política anterior é permitir que o usuário execute qualquer instância do tipo `t1`, `t2` ou `m3`. No entanto, iniciar uma instância na prática requer acesso a muitos recursos, além da própria instância; por exemplo, imagens, pares de chaves, grupos de segurança, entre outros. A instrução completa é avaliada em relação a cada recurso necessário para executar a instância. Esses recursos adicionais não têm a `ec2:InstanceType` chave de condição, de modo que a verificação `StringLike` falha e o usuário não é concedido a capacidade de executar qualquer tipo de instância.

Para resolver isso, use o operador de condição `StringLikeIfExists`. Dessa forma, o teste só acontece se a chave de condição existir. Você pode ler a política a seguir como: “Se o recurso que está sendo verificado tiver uma chave de condição ‘`ec2:InstanceType`‘, permita a ação apenas se o valor de chave começar com `t1.`, `t2.` ou `m3.`. Se o recurso que está sendo verificado não tiver essa chave de condição, não se preocupe com isso." O asterisco (*) nos valores da chave de condição, quando usado com o operador de condição `StringLikeIfExists`, é interpretado como

um curinga para obter correspondências parciais de strings. A instrução `DescribeActions` inclui as ações necessárias para visualizar a instância no console.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RunInstance",
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
        "StringLikeIfExists": {
          "ec2:InstanceType": [
            "t1.*",
            "t2.*",
            "m3.*"
          ]
        }
      }
    },
    {
      "Sid": "DescribeActions",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeImages",
        "ec2:DescribeInstances",
        "ec2:DescribeVpcs",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - t1.micro`	correspondência
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	`ec2:InstanceType: - m2.micro`	Nenhuma correspondência
`"StringLikeIfExists": { "ec2:InstanceType": [ "t1.*", "t2.*", "m3.*" ] }`	Nenhum `ec2:InstanceType` no contexto da solicitação.	correspondência

Operador de condição para verificar a existência de chaves de condição

Use um operador de condição `Null` para verificar se uma chave de condição não está presente no momento da autorização. Na instrução de política, use `true` (a chave não existe, é nulo) ou `false` (a chave existe e seu valor não é nulo).

Você não pode usar uma variável de política com o operador de condição `Null`.

Por exemplo, é possível usar esse operador de condição para determinar se um usuário está usando credenciais temporárias ou suas próprias credenciais para fazer uma solicitação. Se o usuário estiver usando credenciais temporárias, a chave `aws:TokenIssueTime` existe e tem um valor. O exemplo a seguir mostra uma condição que afirma que o usuário deve usar credenciais temporárias (a chave não pode estar ausente) para que o usuário use a API do Amazon EC2.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"false"}}
  }
}
```

Condições com múltiplas chaves ou valores de contexto

Você pode usar o elemento `Condition` de uma política para testar múltiplas chaves de contexto ou múltiplos valores para uma única chave de contexto em uma solicitação. Quando você faz uma solicitação para a AWS, de forma programada ou pelo Console de gerenciamento da AWS, ela inclui informações sobre o seu principal, operação, tags e muito mais. Use chaves de contexto para testar os valores das correspondentes na solicitação, com as chaves de contexto especificadas na condição da política. Para obter mais sobre informações e saber mais sobre os dados incluídos em uma solicitação, consulte O contexto da solicitação.

Tópicos

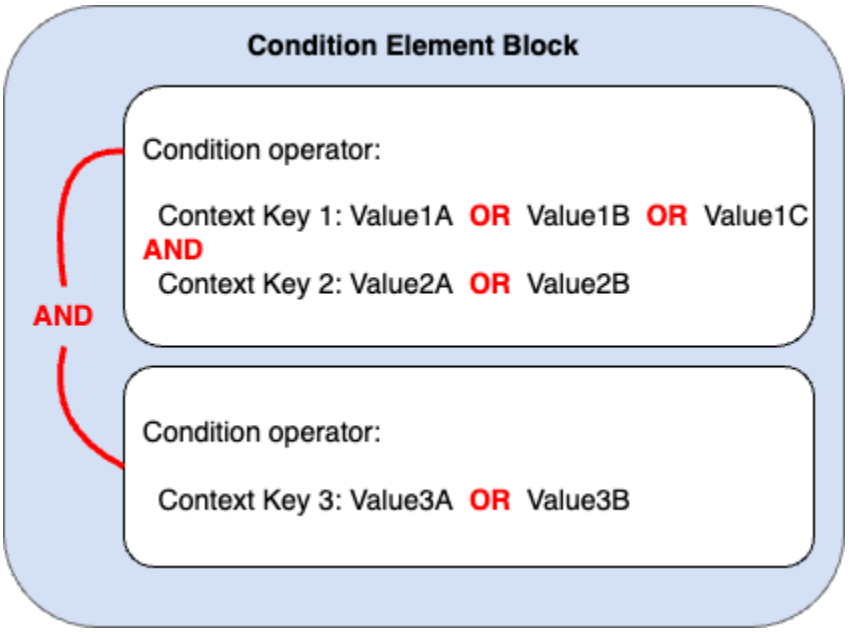
- Lógica de avaliação para múltiplas chaves de contexto ou valores
- Lógica de avaliação para os operadores de conjuntos de condições com correspondência negada

Lógica de avaliação para múltiplas chaves de contexto ou valores

Um elemento `Condition` pode conter múltiplos operadores de condição, e cada operador de condição pode conter múltiplos pares de chave-valor de contexto. A maioria das chaves de contexto oferece suporte ao uso de múltiplos valores, a menos que especificado de outra forma.

- Se a instrução de política tiver múltiplos operadores de condição, os operadores de condição serão avaliados usando uma lógica AND.
- Se sua instrução de política tiver múltiplas chaves de contexto anexadas a um único operador de condição, as chaves de contexto serão avaliadas usando um AND lógico.
- Se um único operador de condição incluir múltiplos valores para uma chave de contexto, esses valores serão avaliados usando um OR lógico.
- Se um único operador de condição com correspondência negada incluir múltiplos valores para uma chave de contexto, esses valores serão avaliados usando um NOR lógico.

Todas as chaves de contexto de um bloco de elementos de condição devem ser resolvidas como verdadeiro para invocar o efeito `Allow` ou `Deny` desejado. A figura a seguir ilustra a lógica de avaliação de uma condição com múltiplos operadores de condição e pares de chave-valor de contexto.



Por exemplo, a política de bucket do S3 a seguir ilustra como a figura anterior é representada em uma política. O bloco de condições inclui os operadores condicionais `StringEquals` e `ArnLike` e as chaves de contexto `aws:PrincipalTag` e `aws:PrincipalArn`. Para invocar o efeito `Allow` ou `Deny` desejado, todas as chaves de contexto de um bloco de condições devem ser resolvidas como verdadeiro. O usuário que faz a solicitação deve ter ambas as chaves de tag da entidade principal, departamento e função, que incluem um dos valores de chave de tag especificados na política. Além disso, o ARN da entidade principal do usuário que faz a solicitação deve corresponder a um dos valores `aws:PrincipalArn` especificados na política a serem avaliados como verdadeiro.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`	correspondência
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": {`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/ Nikki`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
"aws:PrincipalArn": ["arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary"] }		
"StringEquals": { "aws:PrincipalTag/department": ["finance", "hr", "legal"], "aws:PrincipalTag/role": ["audit", "security"] }, "ArnLike": { "aws:PrincipalArn": ["arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary"] }	aws:PrincipalTag/department: hr aws:PrincipalTag/role: *payroll* aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado

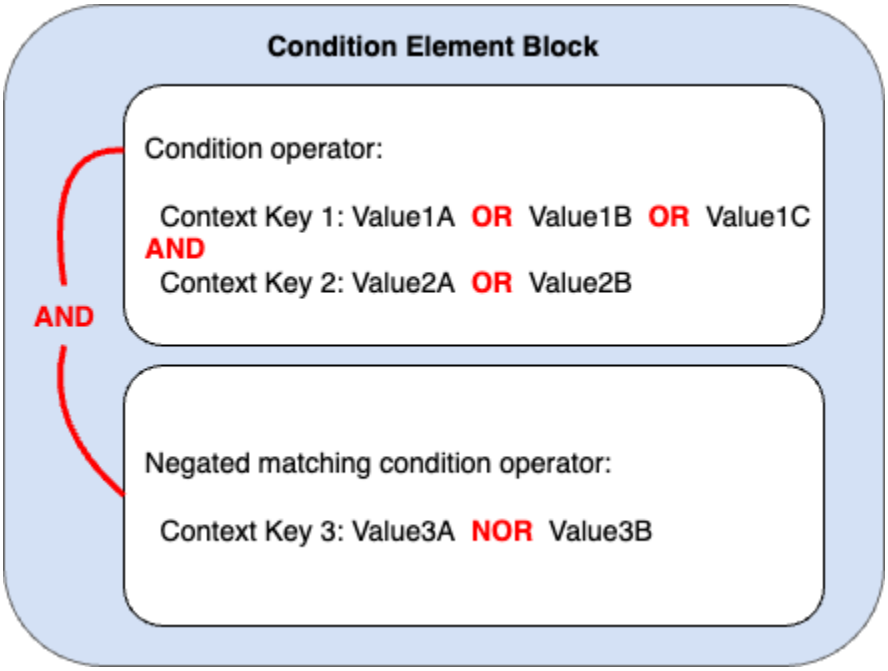
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Nenhum `aws:PrincipalTag/role` no contexto da solicitação.
	`aws:PrincipalTag/department: hr`
	`aws:PrincipalArn: arn:aws:iam::222222222222:user/Mary`
		Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Nenhum `aws:PrincipalTag` no contexto da solicitação. `aws:PrincipalArn: arn:aws:iam::22222222222222:user/Mary`	Nenhuma correspondência

Lógica de avaliação para os operadores de conjuntos de condições com correspondência negada

Alguns operadores de condição, como `StringNotEquals` ou `ArnNotLike`, usam a correspondência negada para comparar os pares de chave-valor de contexto em sua política com os pares de chave-valor de contexto em uma solicitação. Quando múltiplos valores são especificados para uma chave de contexto única em uma política com operadores de condição de correspondência negados, as permissões efetivas funcionam como um `NOR` lógico. Na correspondência negada, um `NOR` ou `NOT OR` lógico retornará verdadeiro somente se todos os valores forem avaliados como falsos.

A figura a seguir ilustra a lógica de avaliação de uma condição com múltiplos operadores de condição e pares de chave-valor de contexto. A figura contém um operador de condição de correspondência negado para a chave de contexto 3.



Por exemplo, a política de bucket do S3 a seguir ilustra como a figura anterior é representada em uma política. O bloco de condições inclui os operadores condicionais `StringEquals` e `ArnNotLike` e as chaves de contexto `aws:PrincipalTag` e `aws:PrincipalArn`. Para invocar o efeito `Allow` ou `Deny` desejado, todas as chaves de contexto de um bloco de condições devem ser resolvidas como verdadeiro. O usuário que faz a solicitação deve ter ambas as chaves de tag da entidade principal, departamento e função, que incluem um dos valores de chave de tag especificados na política. Como o operador de condição `ArnNotLike` usa correspondência negada, o ARN da entidade principal do usuário que faz a solicitação não deve corresponder a nenhum dos valores `aws:PrincipalArn` especificados na política a serem avaliados como verdadeiros.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] },`	`aws:PrincipalTag/department: legal aws:PrincipalTag/role: audit aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	correspondência

Condição da política	Contexto da solicitação	Resultado

"ArnNotLike": {
"aws:PrincipalArn":
[
"arn:aws:
iam::222222222222:
user/Ana",
"arn:aws:
iam::222222222222:
user/Mary"
]
}		

"StringEquals": {
"aws:PrincipalTag/
department": [
"finance",
"hr",
"legal"
],
"aws:PrincipalTag/
role": [
"audit",
"security"
]
},
"ArnNotLike": {
"aws:PrincipalArn":
[
"arn:aws:
iam::222222222222:
user/Ana",
"arn:aws:
iam::222222222222:
user/Mary"
]
}
	aws:PrincipalTag/d
	epartment: hr
	aws:PrincipalTag/role:
	audit
	aws:PrincipalArn:
	arn:aws:iam::22222
	2222222:user/ *Mary*
		Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	`aws:PrincipalTag/department: hr aws:PrincipalTag/role: payroll aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	**>** Nenhum `aws:PrincipalTag/role` no contexto da solicitação. `aws:PrincipalTag/department: hr aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:PrincipalTag/department": [ "finance", "hr", "legal" ], "aws:PrincipalTag/role": [ "audit", "security" ] }, "ArnNotLike": { "aws:PrincipalArn": [ "arn:aws:iam::222222222222:user/Ana", "arn:aws:iam::222222222222:user/Mary" ] }`	Nenhum `aws:PrincipalTag` no contexto da solicitação. `aws:PrincipalArn: arn:aws:iam::222222222222:user/Nikki`	Nenhuma correspondência

Chaves de contexto de valor único vs. de múltiplos valores

A diferença entre as chaves de contexto de valor único e as de múltiplos valores depende do número de valores do contexto da solicitação, e não do número de valores na condição da política.

- Chaves de contexto de condição de valor único têm no máximo um valor no contexto de autorização. Por exemplo, quando você marca recursos na AWS, cada tag de recurso é armazenada como um par de chave-valor. Como uma chave de tag de recurso só pode ter um único valor de tag, the section called “ResourceTag” é uma chave de contexto de valor único. Não use um operador de conjunto de condição com uma chave de condição de valor único.
- As chaves de contexto de condição de múltiplos valores podem ter múltiplos valores no contexto da solicitação. Por exemplo, ao marcar recursos na AWS, você pode incluir vários pares de chave-

valor de tag em uma única solicitação. Portanto, the section called “TagKeys” é uma chave de contexto de valores múltiplos. As chaves de contexto de múltiplos valores exigem um operador de conjunto de condições.

Por exemplo, uma solicitação pode ser originada em até um endpoint da VPC. Logo, the section called “SourceVpce” é uma chave de contexto de valor único. Como um serviço pode ter mais de um nome de entidade principal de serviço pertencente ao serviço, the section called “PrincipalServiceNamesList” é uma chave de contexto de múltiplos valores.

> **Important**
>
> A diferença entre as chaves de contexto de valor único e as de múltiplos valores depende do número de valores do contexto da solicitação, não do número de valores na condição da política.

Para exemplos de políticas que demonstram condições com múltiplas chaves e valores de contexto, consulte Exemplos de políticas de condições.

Principais pontos

- As classificações Valor único e Valores múltiplos constam na descrição de cada chave de contexto de condição como Tipo de valor no tópico AWSChaves de contexto de condições globais da.
- As chaves de contexto de vários valores presentes na referência de autorização de serviço são representadas por um prefixo `ArrayOf` seguido pelo tipo da categoria do operador de condição, como `ArrayOfString` ou `ArrayOfARN`, o que indica que a solicitação pode conter vários valores para uma chave de contexto de condição.
- É possível usar qualquer chave de contexto de valor único disponível como variável de política, mas não é possível fazer isso com uma chave de contexto de múltiplos valores. Para obter mais informações sobre variáveis de política, consulte Elementos de política do IAM: variáveis e etiquetas.
- Ao usar chaves de contexto que incluem pares de chave-valor, é importante observar que, embora possa haver vários valores de tag-chave, cada `tag-key` pode ter apenas um valor.
  - As chaves de contexto the section called “PrincipalTag”, the section called “RequestTag” e the section called “ResourceTag” são classificadas como de valor único.
  - A chave the section called “TagKeys” define quais chaves de etiquetas são permitidas em uma solicitação, mas não inclui os valores dessas chaves de etiquetas. Como é possível incluir vários

pares de chave-valor de etiquetas em uma solicitação, `aws:TagKeys` é uma chave de contexto de vários valores.

- As chaves de contexto de múltiplos valores exigem um operador de conjunto de condições. Não use operadores de conjuntos de condição `ForAllValues` ou `ForAnyValue` com chaves de condição de valor único. O uso de operadores de conjunto de condições com chaves de contexto de valor único pode levar a políticas excessivamente permissivas.

Operadores de conjunto para chaves de contexto de vários valores

Para comparar sua chave de contexto de condição com uma chave de contexto de solicitação com múltiplos valores de chave, você deve usar os operadores de conjunto `ForAllValues` ou `ForAnyValue`. Esses operadores de conjuntos são usados para comparar dois conjuntos de valores, como o conjunto de etiquetas em uma solicitação e o conjunto de etiquetas em uma condição de política.

Os qualificadores `ForAllValues` e `ForAnyValue` adicionam a funcionalidade de operação de conjunto ao operador da condição, permitindo que você teste chaves de contexto da solicitação em relação a múltiplos valores de chave de contexto em uma condição de política. Além disso, se você incluir uma chave de contexto de string de múltiplos valores na política com um curinga ou uma variável, também deverá usar o operador de condição `StringLike`. Os diferentes valores de chaves de condição devem ser colocados entre colchetes, como em uma matriz, por exemplo `"Key2":["Value2A", "Value2B"]`.

ForAllValues

O qualificador `ForAllValues` realiza testes para verificar se o valor de cada elemento no contexto da solicitação corresponde ao operador de condição que o segue. A condição retornará `true` se todos os valores das chaves de contexto presentes na solicitação corresponderem aos valores para as chaves de contexto definidos na política. Além disso, a condição retornará `true` se não houver chaves de contexto presentes na solicitação.

> **Important**
>
> Tenha cuidado ao usar `ForAllValues` em políticas com efeito `Allow`, pois isso pode resultar em permissões excessivas se a ausência de chaves de contexto no contexto da solicitação não for prevista. Sempre inclua o operador de condição `Null` com um valor

`false` em sua política para verificar se a chave de contexto existe e se seu valor não é nulo. Para ver um exemplo, consulte Controlar o acesso com base em chaves de tag.

Exemplo do operador de conjunto ForAllValues

No exemplo apresentado a seguir, o operador ForAllValues é usado com aws:TagKeys para permitir que os usuários excluam etiquetas específicas atribuídas a uma instância do EC2. Esta política permite a exclusão apenas das etiquetas `environment` e `cost-center` por parte dos usuários. Essas etiquetas podem ser excluídas separada ou simultaneamente. As chaves de etiquetas presentes na solicitação devem corresponder exatamente às chaves especificadas na política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                },
                "Null": {
                    "aws:TagKeys": "false"
                }
            }
        }
    ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - environment`	correspondência
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - cost-center`	correspondência
`"ForAllValues:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }, "Null": { "aws:TagKeys": "false" }`	`aws:TagKeys: - environment - cost-center`	correspondência
`"ForAllValues:StringEquals": {`	`aws:TagKeys: - environment`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado

<pre>"aws:TagKeys": [
"environment",
"cost-center"
]
},
"Null": {
"aws:TagKeys":
"false"
}</pre>
	<pre>- dept</pre>	

<pre>"ForAllValues:Stri
ngEquals": {
"aws:TagKeys": [
"environment",
"cost-center"
]
},
"Null": {
"aws:TagKeys":
"false"
}</pre>
	Nenhum `aws:TagKeys` no contexto da solicitação.	Nenhuma correspondência

Vale destacar que, no exemplo anterior, o resultado é “Nenhuma correspondência” devido à verificação da condição “Null”, que bloqueia a correspondência quando a chave de contexto não está presente. Essa é uma prática recomendada para evitar políticas excessivamente permissivas.

ForAnyValue

O qualificador `ForAnyValue` realiza testes para verificar se ao menos um dos valores presentes no conjunto de chaves de contexto da solicitação corresponde a algum dos valores de chaves de contexto definidos na condição da política. A condição retornará `true` se algum valor de chave de contexto presente na solicitação corresponder a um valor para as chaves de contexto definido na política. Se não houver correspondência entre as chaves de contexto ou se a chave não existir, a condição retornará `false`.

> **Important**
>
> Ao usar `ForAnyValue` com o efeito `Deny`, se a chave de contexto não estiver presente na solicitação, a política será avaliada como Nenhuma correspondência. Para garantir um comportamento consistente, adicione uma verificação explícita da condição `Null` em sua política para confirmar se a chave de contexto existe. Para obter detalhes, consulte Operador de condição para verificar a existência de chaves de condição.

Exemplo do operador de conjunto ForAnyValue

No exemplo apresentado a seguir, o operador ForAnyValue é usado com aws:TagKeys para permitir que os usuários excluam etiquetas específicas atribuídas a uma instância do EC2. Esta política permite a exclusão de etiquetas, por parte dos usuários, se as chaves de etiquetas especificadas na solicitação incluírem `environment` ou `cost-center`. A solicitação pode conter chaves de etiquetas adicionais, mas deve incluir, no mínimo, uma das chaves especificadas para que a condição seja satisfeita.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                }
            }
        }
    ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment`	correspondência
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - cost-center`	correspondência
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - cost-center`	correspondência
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	`aws:TagKeys: - environment - dept`	correspondência
`"ForAnyValue:StringEquals": {`	`aws:TagKeys:`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"aws:TagKeys": [ "environment", "cost-center" ] }`	`- dept`	
`"ForAnyValue:StringEquals": { "aws:TagKeys": [ "environment", "cost-center" ] }`	Nenhum `aws:TagKeys` no contexto da solicitação.	Nenhuma correspondência

Exemplos de políticas de condições

Nas políticas do IAM, é possível especificar múltiplos valores para chaves de contexto de valor único e de múltiplos valores para comparação com o contexto da solicitação. O conjunto de exemplos de políticas a seguir demonstra condições de política com múltiplos valores e chaves de contexto.

> **Note**
>
> Se você quiser enviar uma política para ser incluída neste guia de referência, use o botão Feedback na parte inferior desta página. Para obter exemplos de políticas baseadas em identidade do IAM, consulte Exemplos de políticas baseadas em identidade do IAM.

Exemplos de políticas de condição: chaves de contexto de valor único

- Vários blocos de condição com chaves de contexto de valor único
- Um bloco de condição com múltiplos valores e chaves de contexto de valor único

Exemplos de políticas de condição: chaves de contexto de múltiplos valores

- Política de negação com operador de conjunto de condições ForAllValues
- Política de negação com operador de conjunto de condições ForAnyValue

Exemplos de chave de contexto de múltiplos valores

O conjunto de exemplos de políticas a seguir demonstra como criar condições de política com chaves de contexto de múltiplos valores.

Exemplo: política de negação com operador de conjunto de condições ForAllValues

Os exemplos a seguir mostram como usar uma política baseada em identidade para negar o uso de ações de marcação do IAM quando prefixos de chave de marcação específicos são incluídos na solicitação. Os valores para `aws:TagKeys` inclua um curinga (*) para correspondência parcial de string. A política inclui o operador de conjunto `ForAllValues` com chave de contexto `aws:TagKeys` porque a chave de contexto da solicitação pode conter múltiplos valores. Para a chave de contexto `aws:TagKeys` ser correspondente, cada valor na solicitação de contexto deve corresponder a pelo menos um valor na política.

O operador de conjunto `ForAllValues` também retornará verdadeiro se não houver chaves de contexto na solicitação.

Você pode evitar que chaves de contexto ausentes ou chaves de contexto com valores vazios sejam avaliadas como verdadeiro, ao incluir o operador de condição `Null` em sua política com um valor `false` para verificar se a chave de contexto da solicitação existe e se seu valor não é nulo. Para obter mais informações, consulte Operador de condição para verificar a existência de chaves de condição.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

Example Negar um único valor de condição de política para uma chave de contexto de vários valores

No exemplo a seguir, a política nega solicitações onde os valores para `aws:TagKeys` na solicitação não incluam o prefixo key1. O contexto da solicitação pode ter vários valores, mas devido ao operador do conjunto de condição `ForAllValues`, todos os valores da chave de tag no contexto da solicitação devem começar com o prefixo key1.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": "key1*"
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação. Para uma Declaração de negação, a Correspondência é Negada e Nenhuma correspondência é Não negada, portanto, isso pode ser permitido por outra declaração.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:legal`	Nenhuma correspondência Pode ser permitido por outra declaração.
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key1:hr - key1:personnel`	Nenhuma correspondência Pode ser permitido por outra declaração.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	`aws:TagKeys: - key2:audit`	correspondência
`"ForAllValues:StringNotLike": { "aws:TagKeys": "key1*" }`	Nenhum `aws:TagKeys` no contexto da solicitação.	correspondência

Example Negar vários valores de condição de política para uma chave de contexto de vários valores

No exemplo a seguir, a política nega solicitações onde os valores para `aws:TagKeys` na solicitação não incluam o prefixo key1 ou key2. O contexto da solicitação pode ter vários valores, mas devido ao operador do conjunto de condição `ForAllValues`, todos os valores da chave de tag no contexto da solicitação devem começar com o prefixo key1 ou key2.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "ForAllValues:StringNotLike": {
          "aws:TagKeys": [
```

```
            "key1*",
            "key2*"
          ]
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação. Para uma Declaração de negação, a Correspondência é Negada e Nenhuma correspondência é Não negada, portanto, isso pode ser permitido por outra declaração.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:legal`	Nenhuma correspondência Pode ser permitido por outra declaração.
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:hr - key1:personnel`	Nenhuma correspondência Pode ser permitido por outra declaração.
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key1:hr - key2:audit`	Nenhuma correspondência Pode ser permitido por outra declaração.

Condição da política	Contexto da solicitação	Resultado
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	`aws:TagKeys: - key3:legal`	correspondência
`"ForAllValues:StringNotLike": { "aws:TagKeys": [ "key1*", "key2*" ] }`	Nenhum `aws:TagKeys` no contexto da solicitação.	correspondência

Exemplo: política de negação com operador de conjunto de condições ForAnyValue

O exemplo de política baseada em identidade a seguir negará a criação de snapshots de volumes de instâncias do EC2 se algum snapshot estiver marcado com uma das chaves de etiqueta especificadas na política, `environment` ou `webserver`. A política inclui o operador de conjunto `ForAnyValue` com chave de contexto `aws:TagKeys` porque a chave de contexto da solicitação pode conter múltiplos valores. Se sua solicitação de marcação incluir qualquer chave-valor de etiqueta especificado na política, a chave de contexto `aws:TagKeys` retornará como verdadeiro, invocando o efeito da política de negação.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "ec2:CreateSnapshot",
        "ec2:CreateSnapshots"
      ],
      "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:TagKeys": "webserver"
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação. Para uma Declaração de negação, a Correspondência é Negada e Nenhuma correspondência é Não negada, portanto, isso pode ser permitido por outra declaração.

Condição da política	Contexto da solicitação	Resultado
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - webserver`	correspondência
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	`aws:TagKeys: - environment - webserver - test`	correspondência
`"ForAnyValue:StringEquals": {`	`aws:TagKeys: - environment - test`	Nenhuma correspondência

Condição da política	Contexto da solicitação	Resultado
`"aws:TagKeys": "webserver" }`		Pode ser permitido por outra declaração.
`"ForAnyValue:StringEquals": { "aws:TagKeys": "webserver" }`	Nenhum `aws:TagKeys` no contexto da solicitação.	Nenhuma correspondência Pode ser permitido por outra declaração.

Exemplos de políticas de chave de contexto de valor único

O conjunto de exemplos de políticas a seguir demonstra como criar condições de política com chaves de contexto de valor único.

Exemplo: múltiplos blocos de condição com chaves de contexto de valor único

Quando um bloco de condição tem várias condições, cada uma com uma única chave de contexto, todas as chaves de contexto devem ser resolvidas como verdadeiras para o efeito desejado `Allow` ou `Deny` a ser invocado. Quando você usa operadores de condição de correspondência negados, inverte-se a lógica de avaliação do valor da condição.

O exemplo a seguir permite que os usuários criem volumes do EC2 e apliquem qualquer etiqueta aos volumes durante a criação do volume. O contexto da solicitação deve incluir um valor para a chave de contexto `aws:RequestTag/project`, e o valor da chave de contexto `aws:ResourceTag/environment` pode ser qualquer coisa, exceto produção.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
```

```
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:us-east-1:123456789012:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

O contexto da solicitação deve incluir o par etiqueta-valor do projeto e não pode ser criado para que um recurso de produção invoque o efeito `Allow`. O volume do EC2 a seguir foi criado com êxito porque o nome do projeto é `Feature3` com uma etiqueta de recurso `QA`.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Exemplo: um bloco de condição com múltiplos valores e chaves de contexto de valor único

Quando um bloco de condição contém múltiplas chaves de contexto e cada chave de contexto tem múltiplos valores, cada chave de contexto deve ser resolvida como verdadeiro para pelo menos um valor de chave para o efeito desejado `Allow` ou `Deny` a ser invocado. Quando você usa operadores de condição de correspondência negados, inverte-se a lógica de avaliação do valor da chave de contexto.

Com o exemplo a seguir, os usuários podem iniciar e executar tarefas nos clusters do Amazon Elastic Container Service.

- O contexto da solicitação deve incluir `production` OR `prod-backup` para a chave de contexto `aws:RequestTag/environment` AND.
- A chave de contexto `ecs:cluster` garante que as tarefas sejam executadas em qualquer um dos clusters ARN `default1` OR `default2` do ECS.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

A tabela a seguir mostra como a AWS avalia essa política com base nos valores de chave de condição da solicitação.

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: environment:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default1`	correspondência
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: environment:prod-backup ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default2`	correspondência

Condição da política	Contexto da solicitação	Resultado
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	`aws:RequestTag: webserver:production ecs:cluster: arn:aws:ecs:us-east-1:111122223333:cluster/default2`	Nenhuma correspondência
`"StringEquals": { "aws:RequestTag/environment": [ "production", "prod-backup" ] }, "ArnEquals": { "ecs:cluster": [ "arn:aws:ecs:us-east-1:111122223333:cluster/default1", "arn:aws:ecs:us-east-1:111122223333:cluster/default2" ] }`	Nenhum `aws:RequestTag` no contexto da solicitação. `ecs:cluster arn:aws:ecs:us-east-1:111122223333:cluster/default2`	Nenhuma correspondência

Elementos de política do IAM: variáveis e etiquetas

Use variáveis de política do AWS Identity and Access Management (IAM) como espaços reservados quando você não souber o valor exato de um recurso ou uma chave de condição ao escrever a política.

> **Note**
>
> Se a AWS não puder resolver uma variável, isso poderá invalidar toda a instrução. Por exemplo, se você usar a variável `aws:TokenIssueTime`, a variável será resolvida como um valor somente quando o solicitante tiver sido autenticado com credenciais temporárias (uma função do IAM). Para evitar que as variáveis provoquem instruções inválidas, use o operador de condição ...IfExists.

Tópicos

- Introdução
- Usar variáveis em políticas
- Tags como variáveis de política
- Onde você pode usar variáveis de política
- Variáveis de política sem valor
- Solicitar informações que você pode usar para variáveis de política
- Especificação de valores padrão
- Para obter mais informações

Introdução

Nas políticas do IAM, muitas ações permitem que você forneça um nome para os recursos específicos cujo acesso deseja controlar. For exemplo, a política a seguir permite que o usuário liste, leia e grave objetos no bucket do S3 `amzn-s3-demo-bucket` para projetos de `marketing`.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/marketing/*"]
    }
  ]
}
```

Em alguns casos, talvez você saiba o nome exato do recurso ao elaborar a política. Talvez você queira generalizar a política para que ele funcione para muitos usuários sem a necessidade de fazer uma cópia exclusiva da política para cada usuário. Em vez de criar uma política específica para cada usuário, recomendamos criar uma única política de grupo que funcione para qualquer usuário desse grupo.

Usar variáveis em políticas

É possível definir valores dinâmicos dentro das políticas usando variáveis de política que definem espaços reservados em uma política.

As variáveis são marcadas usando um prefixo **$** seguido por um par de chaves (**{ }**) que incluem o nome da variável do valor da solicitação.

Quando a política for avaliada, suas variáveis serão substituídas por valores provenientes das chaves de contexto condicionais passadas na solicitação. As variáveis podem ser usadas em políticas baseadas em identidade, políticas de recursos, políticas de controle de serviços, políticas de sessão e políticas de endpoint da VPC. As políticas baseadas em identidade usadas como limites de permissões também são compatíveis com variáveis de política.

As chaves de contexto de condição globais podem ser usadas como variáveis em solicitações entre serviços da AWS. As chaves de condição específicas do serviço também podem ser usadas como variáveis ao interagir com recursos da AWS, mas só estão disponíveis quando as solicitações

são baseadas em recursos compatíveis com elas. Para obter uma lista das chaves de contexto disponíveis para cada serviço e recurso da AWS, consulte a Referência de autorização de serviço. Em certas circunstâncias, não é possível preencher chaves de contexto de condição globais com um valor. Para saber mais, consulte Chaves de contexto de condição globais da AWS.

> **Important**
>
> - Os nomes das chaves não diferenciam maiúsculas de minúsculas. Por exemplo, `aws:CurrentTime` equivale a `AWS:currenttime`.
> - É possível usar qualquer chave de condição de valor único disponível como uma variável. Você não pode usar uma chave de condição de valores múltiplos como uma variável.

O exemplo a seguir mostra uma política para um perfil ou usuário do IAM que substitui um nome do recurso específico por uma variável de política. Você pode reutilizar essa política aproveitando a chave de condição `aws:PrincipalTag`. Quando esta política é avaliada, `${aws:PrincipalTag/team}` permitirá ações somente se o nome do bucket terminar com o nome da equipe da tag `team` da entidade principal.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/team}/*"]
    }
  ]
```

```
}
```

A variável é marcada usando um $ prefixo seguido por um par de chaves ({ }). Dentro dos caracteres ${ }, você pode incluir o nome do valor da solicitação que você deseja usar na política. Os valores que você pode usar serão abordados posteriormente nesta página.

Para obter uma lista desta chave de condição global, consulte aws:PrincipalTag/tag-key na lista de chaves condição globais.

> **Note**
>
> Para usar variáveis de política, você deve incluir o elemento `Version` em uma instrução, e a versão deve ser definida como uma versão que oferece suporte às variáveis de política. As variáveis foram apresentadas na versão `2012-10-17`. Versões anteriores da linguagem da política não são compatíveis com variáveis de política. Se você não incluir o elemento `Version` e defini-lo como uma data de versão apropriada, variáveis como `${aws:username}` serão tratadas como strings literais na política.
> Um elemento de política `Version` é diferente de uma versão de política. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. Uma versão da política, por outro lado, é criada quando você altera uma política gerenciada pelo cliente no IAM. A política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. Para saber mais sobre o elemento de política `Version`, consulte the section called “Version”. Para saber mais sobre as versões de política, consulte the section called “Versionamento de políticas do IAM”.

Uma política que permite que uma entidade principal obtenha objetos do caminho /David de um bucket do S3 tem a seguinte aparência:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/David/*"
      ]
    }
  ]
}
```

Se essa política estiver anexada ao usuário `David`, esse usuário terá objetos do seu próprio bucket do S3, mas você teria que criar uma política específica para cada usuário que incluísse o nome do usuário. Você pode anexar cada política aos usuários individuais.

Ao usar uma variável de política, você pode criar políticas que podem ser reutilizadas. A política a seguir permite que um usuário obtenha objetos de um bucket do Amazon S3 se o valor da chave de tag `aws:PrincipalTag` corresponder ao valor da chave de tag `owner` passado na solicitação.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Ao usar uma variável de política para um usuário dessa forma, você não precisa ter uma política específica para cada usuário individual. No exemplo a seguir, a política é anexada a um perfil do IAM que é assumido pelos gerentes de produto usando credenciais de segurança temporárias. Quando

um usuário faz uma solicitação para adicionar um objeto do Amazon S3, o IAM substituísse o valor da tag `dept` da solicitação atual para a variável `${aws:PrincipalTag}` e avalia a política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOnlyDeptS3Prefix",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/${aws:PrincipalTag/dept}/*"
            ]
        }
    ]
}
```

Tags como variáveis de política

Em alguns serviços da AWS que você pode anexar os próprios atributos personalizados a recursos criados por esses serviços. Por exemplo, você pode aplicar etiquetas a buckets do Amazon S3 ou a usuários do IAM. Essas tags são pares de chave-valor. Você define o nome da chave de tag e o valor associado ao nome dessa chave. Por exemplo, convém criar uma tag com uma chave **`department`** e um valor **`Human Resources`**. Para obter mais informações sobre como etiquetar entidades do IAM, consulte Tags para recursos do AWS Identity and Access Management. Para obter informações sobre como marcar recursos criados por outros serviços da AWS, consulte a documentação desse serviço. Para obter informações sobre como usar o Tag Editor, consulte Trabalhar com o Tag Editor no Console de gerenciamento da AWS Guia do usuário.

Você pode etiquetar recursos do IAM para simplificar a descoberta, a organização e o monitoramento de seus recursos do IAM. Você também pode etiquetar as identidades do IAM para controlar o acesso a recursos ou etiquetar a si mesmo. Para saber mais sobre como usar tags para controlar o acesso, consulte Controle de acesso para usuários e funções do IAM usando etiquetas.

Onde você pode usar variáveis de política

É possível usar variáveis de política no elemento `Resource` e em comparações de string no elemento `Condition`.

Elemento de recursos

É possível usar uma variável de política no elemento `Resource`, mas somente na parte de recurso do ARN. Essa parte do ARN aparece após o quinto sinal de dois pontos (:). Não é possível usar uma variável para substituir partes do ARN antes do quinto sinal de dois pontos, como o serviço ou a conta. Para obter mais informações sobre o formato do ARN, consulte ARNs do IAM.

Para substituir parte de um ARN com um valor de tag, coloque o prefixo e o nome da chave com `${ }`. Por exemplo, o elemento de Recurso a seguir se refere apenas a um bucket com o mesmo nome do valor na tag do usuário de solicitação.

```
"Resource": ["arn:aws::s3:::amzn-s3-demo-bucket/
${aws:PrincipalTag/department}"]
```

Muitos recursos da AWS usam ARNs que contêm um nome criado pelo usuário. A política do IAM a seguir garante que somente os usuários pretendidos com valores de tag access-project, access-application e access-environment correspondentes possam modificar seus recursos. Além disso, usando correspondências curingas *, eles podem permitir sufixos de nomes de recursos personalizados.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
```

```
  ]
}
```

Elemento de condição

É possível usar uma variável de política para valores `Condition` em qualquer condição que envolva os operadores de string ou os operadores de ARN. Os operadores de string incluem `StringEquals`, `StringLike` e `StringNotLike`. Os operadores de ARN incluem `ArnEquals` e `ArnLike`. Não é possível usar uma variável de política com outros operadores, como os operadores `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address` ou `Null`. Para obter mais informações sobre operadores de condição, consulte Elementos de política JSON do IAM: operadores de condição.

Ao fazer referência a uma tag em uma expressão do elemento `Condition`, use o prefixo relevante e o nome da chave de condição. Em seguida, use o valor que você deseja testar no valor da condição.

Por exemplo, o exemplo de política a seguir permite acesso total aos usuários, mas apenas se a tag `costCenter` estiver anexada ao usuário. A tag também deve ter um valor de `12345` ou `67890`. Se a tag não tiver valor, ou qualquer outro valor, a solicitação falhará.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Variáveis de política sem valor

Quando as variáveis de política fazem referência a uma chave de contexto de condição que não tem valor ou não está presente em um contexto de autorização para uma solicitação, o valor é efetivamente nulo. Não há valor igual ou semelhante. As chaves de contexto de condição podem não estar presentes no contexto de autorização quando:

- Você está usando chaves de contexto de condição específicas do serviço em solicitações a recursos que não oferecem suporte a essa chave de condição.
- As etiquetas em entidades principais, sessões, recursos ou solicitações do IAM não estão presentes.
- Outras circunstâncias, conforme listadas para cada chave de contexto de condição global em AWSChaves de contexto de condições globais da.

Quando você usa uma variável sem valor no elemento de condição de uma política do IAM, Elementos de política JSON do IAM: operadores de condição como `StringEquals` ou `StringLike` não tem correspondência, e a instrução de política não entra em vigor.

Operadores de condição invertida como `StringNotEquals` ou `StringNotLike` correspondem a um valor nulo, pois o valor da chave de condição que serve de base para o teste não é igual ou semelhante ao valor efetivamente nulo.

Neste exemplo, `aws:principaltag/Team` deve ser igual a `s3:ExistingObjectTag/Team` para permitir o acesso. O acesso é explicitamente negado quando não `aws:principaltag/Team` está definido. Se uma variável sem valor no contexto de autorização for usada como parte do elemento `Resource` ou `NotResource` de uma política, o recurso que inclui uma variável de política sem valor não corresponderá a nenhum recurso.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
```

```
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
    }
  ]
}
```

Solicitar informações que você pode usar para variáveis de política

É possível usar o elemento `Condition` de uma política JSON para comparar chaves no contexto da solicitação com os valores de chave especificados na política. Quando você usa uma variável de política, a AWS substitui um valor da chave de contexto de solicitação no lugar da variável na política.

Valores de chave de principal

Os valores de `aws:username`, `aws:userid` e `aws:PrincipalType` dependem do tipo de principal que iniciou a solicitação. Por exemplo, a solicitação pode ser feita usando as credenciais de um usuário do IAM, um perfil do IAM ou o Usuário raiz da conta da AWS. A seguinte de tabelas mostra valores para essas chaves de diferentes tipos de entidades principais.

- Usuário raiz da conta da AWS
  - `aws:username`: (não está presente)
  - `aws:userid`: ID da Conta da AWS
  - `aws:PrincipalType`: `Account`
- Usuário do IAM
  - `aws:username`: *`IAM-user-name`*
  - `aws:userid`: ID exclusivo
  - `aws:PrincipalType`: `User`
- AWS STS Entidade principal de usuário federado do
  - `aws:username`: (não está presente)
  - `aws:userid`: *`account`*:*`caller-specified-name`*
  - `aws:PrincipalType`: `FederatedUser`
- Entidades principais federadas OIDC e do SAML

> **Note**
>
> Para obter informações sobre as chaves de política que estão disponíveis ao usar a federação OIDC, consulte Federação OIDC.

  - `aws:username`: (não está presente)
  - `aws:userid`: (não está presente)
  - `aws:PrincipalType`: `AssumedRole`
- Função assumida
  - `aws:username`: (não está presente)
  - `aws:userid`: *`role-id`*:*`caller-specified-role-name`*
  - `aws:PrincipalType`: `Assumed role`
- Função atribuída a uma instância do Amazon EC2
  - `aws:username`: (não está presente)
  - `aws:userid`: *`role-id`*:*`ec2-instance-id`*
  - `aws:PrincipalType`: `Assumed role`
- Autor da chamada anônimo (somente Amazon SQS, Amazon SNS e Amazon S3)
  - `aws:username`: (não está presente)
  - `aws:userid`: (não está presente)
  - `aws:PrincipalType`: `Anonymous`

Para os itens nesta lista, observe o seguinte:

- not present (não está presente) significa que o valor não está nas informações da solicitação atual, e que qualquer tentativa de correspondê-lo falhará e fará com que a instrução seja invalidada.
- *`role-id`* é um identificador exclusivo atribuído a cada função na criação. É possível exibir o ID da função com o comando da AWS CLI: `aws iam get-role --role-name` *`rolename`*
- *`caller-specified-name`* e *`caller-specified-role-name`* são nomes que são passados pelo processo de chamada (como um aplicativo ou serviço) ao fazer uma chamada para obter credenciais temporárias.

- *ec2-instance-id* é um valor atribuído à instância quando ela é iniciada e aparece na página Instances (Instâncias) do console do Amazon EC2. Também é possível exibir o ID da instância executando o comando da AWS CLI: `aws ec2 describe-instances`

Informações disponíveis em solicitações para entidades principais federadas

Entidades principais federadas são usuários que são autenticados usando um sistema diferente do IAM. Por exemplo, uma empresa pode ter uma aplicação para uso interno que faz chamadas para a AWS. Pode ser impraticável fornecer uma identidade do IAM para cada usuário corporativo que use o aplicativo. Em vez disso, a empresa pode usar um aplicativo proxy (camada intermediária) que tem uma única identidade do IAM ou a empresa pode usar um provedor de identidade (IdP) SAML. O aplicativo proxy ou IdP SAML autentica usuários individuais usando a rede corporativa. Um aplicativo proxy pode usar sua identidade do IAM para obter credenciais de segurança temporárias para usuários individuais. Um IdP SAML pode, efetivamente, trocar informações de identidade por credenciais de segurança temporárias da AWS. As credenciais de segurança temporárias podem ser usadas para acessar os recursos da AWS.

Da mesma forma, você pode criar uma aplicação para um dispositivo móvel no qual ela precise acessar os recursos da AWS. Nesse caso, você pode usar a federação OIDC, em que a aplicação autentica o usuário usando um provedor de identidades conhecido, como Login with Amazon, Amazon Cognito, Facebook ou Google. A aplicação pode então usar as informações de autenticação do usuário desses provedores para obter credenciais de segurança temporárias para acessar recursos da AWS.

A maneira recomendada de usar a federação OIDC é aproveitando as vantagens oferecidas pelo Amazon Cognito e os SDKs móveis da AWS. Para saber mais, consulte:

- Guia do usuário do Amazon Cognito
- Cenários comuns para credenciais temporárias

Caracteres especiais

Há algumas variáveis de políticas predefinidas especiais com valores fixos que permitem representar caracteres que, de outra forma, têm um significado especial. Se esses caracteres especiais fizerem parte da string, você estiver tentando estabelecer uma correspondência e os inseriu literalmente, eles seriam mal-interpretados. Por exemplo, inserir um asterisco * na string seria interpretado como curinga, fazendo correspondência com qualquer caractere, em vez de como um * literal. Nesses casos, você pode usar as seguintes variáveis de política predefinidas:

- ${*} - use quando precisar um asterisco (*).
- ${?} - use quando precisar de um ponto de interrogação (?).
- ${$} - use quando precisar de um caractere de cifrão ($).

Essas variáveis de política predefinidas podem ser usadas em qualquer string onde é possível usar variáveis de política normais.

Especificação de valores padrão

Para adicionar um valor padrão a uma variável, coloque o valor padrão entre aspas simples (`' '`) e separe o texto da variável e o valor padrão com uma vírgula e espaço (`, `).

Por exemplo, se uma entidade principal estiver marcada com um `team=yellow`, é possível acessar o bucket do Amazon S3 da `ExampleCorp's` nomeado como `amzn-s3-demo-bucket-yellow`. Uma política com esse recurso permite que os membros da equipe acessem o bucket da própria equipe, mas não os de outras equipes. Para usuários sem etiquetas de equipe, ele define um valor padrão de `company-wide` para o nome do bucket. Esses usuários podem acessar somente o bucket `amzn-s3-demo-bucket-company-wide`, onde podem visualizar informações gerais, como instruções para ingressar em uma equipe.

```
"Resource":"arn:aws:s3:::amzn-s3-demo-bucket-${aws:PrincipalTag/team, 'company-wide'}"
```

Para obter mais informações

Para obter mais informações sobre políticas, consulte o seguinte:

- Políticas e permissões no AWS Identity and Access Management
- Exemplos de políticas baseadas em identidade do IAM
- Referência de elemento de política JSON do IAM
- Lógica da avaliação de política
- Federação OIDC

Elementos de política JSON do IAM: tipos de dados compatíveis

Esta seção lista os tipos de dados compatíveis quando você especifica valores em políticas JSON. A linguagem da política não é compatível com todos os tipos para cada elemento de política; para obter informações sobre cada elemento, consulte as seções anteriores.

- Strings
- Números (Inteiros e floats)
- Booleano
- Null
- Listas
- Mapas
- Estruturas (que são apenas mapas aninhados)

A tabela a seguir mapeia cada tipo de dado à serialização. Observe que todas as políticas devem estar em UTF-8. Para obter informações sobre os tipos de dados JSON, consulte RFC 4627.

Tipo	JSON
String	String
Inteiro	Número
Float	Número
Booleano	verdadeiro falso
Null	nulo
Data	String aderindo ao perfil W3C da ISO 8601
IpAddress	String aderindo a RFC 4632
Lista	Array
Objeto	Objeto

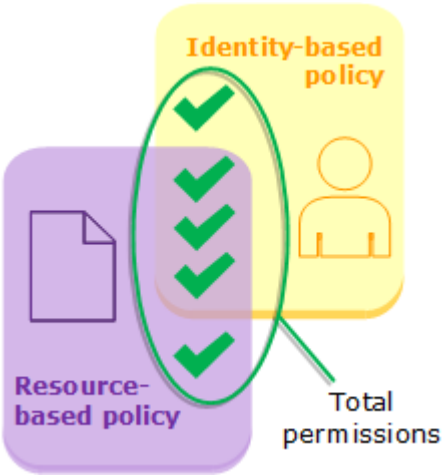
Lógica da avaliação de política

Quando uma entidade principal tenta usar o Console de gerenciamento da AWS, a API da AWS ou a AWS CLI, ela envia uma solicitação para a AWS. Quando um serviço da AWS recebe a solicitação, a AWS realiza várias etapas para determinar se deve permitir ou negar a solicitação.

1. Autenticação: a AWS primeiro autentica a entidade principal que faz a solicitação, se necessário. Essa etapa não é necessária para alguns serviços, como o Amazon S3, que permite algumas solicitações de usuários anônimos.
2. Processamento do contexto da solicitação – AWS a processa as informações coletadas na solicitação para determinar quais políticas se aplicam à solicitação.
3. Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso: a AWS avalia todos os tipos de política, e a ordem das políticas afeta a forma como são avaliadas. A AWS então processa as políticas em relação ao contexto da solicitação para determinar se a solicitação é permitida ou negada.

Avaliação das políticas baseadas em identidade com políticas baseadas em recurso

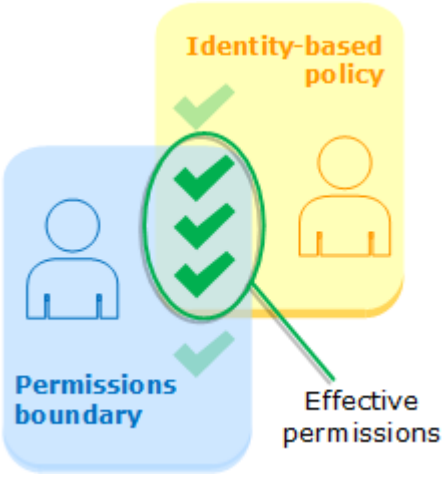
As políticas baseadas em identidade e as baseadas em recurso concedem permissões para as identidades ou recursos aos quais elas estão conectadas. Quando uma entidade do IAM (usuário ou função) solicita acesso a um recurso na mesma conta, a AWS avalia todas as permissões concedidas pelas políticas baseadas em identidade e as políticas baseadas em recurso. As permissões resultantes são a união das permissões dos dois tipos. Se uma ação for permitida por uma política baseada em identidade, uma política baseada em recurso ou ambas, o AWS permitirá a ação. Uma negação explícita em qualquer uma dessas políticas substitui a permissão.



Avaliação das políticas baseadas em identidade com limites de permissões

Quando o AWS avalia as políticas baseadas em identidade e limites de permissões para um usuário, as permissões resultantes são a interseção das duas categorias. Isso significa que, quando você adiciona um limite de permissões a um usuário com políticas baseadas em identidade existentes, você pode reduzir as ações que o usuário pode executar. Como alternativa, quando você remove

o limite de permissões de um usuário, você pode aumentar as ações que ele pode executar. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para visualizar informações sobre como outros tipos de política são avaliadas com limites de permissões, consulte Avaliar permissões efetivas com limites.



Avaliação das políticas baseadas em identidade com SCPs ou RCPs do AWS Organizations

Quando um usuário pertence a uma conta que é membro de uma organização e acessa um recurso que não tenha uma política baseada em recursos configurada, as permissões resultantes são a interseção das políticas do usuário, políticas de controle de serviços (SCPs) e política de controle de recursos (RCP). Isso significa que uma ação deve ser permitida pelos três tipos de política. Uma negação explícita na política baseada em identidade, uma SCP ou uma RCP se sobrepõem à permissão.

É possível saber se sua conta é membro de uma organização no AWS Organizations. Os membros da organização podem ser afetados por uma SCP ou RCP. Para visualizar esses dados usando

o comando da AWS CLI ou operação de API da AWS, você deve ter permissões para a ação `organizations:DescribeOrganization` da sua entidade do AWS Organizations. Você deve ter permissões adicionais para executar a operação no console do AWS Organizations. Para saber se uma SCP ou RCP está negando acesso a uma solicitação específica ou alterar as permissões efetivas, entre em contato com seu administrador do AWS Organizations.

Processamento do contexto da solicitação

Quando a AWS avalia e autoriza uma solicitação, ela reúne as informações da solicitação em um contexto da solicitação. O contexto da solicitação contém todas as informações que podem ser usadas na avaliação de políticas.

- Entidade principal: o usuário, perfil ou entidade principal de usuário federado que enviou a solicitação. As informações sobre a entidade principal incluem as políticas que estão associadas à entidade principal.
- Ações: uma ou mais ações que a entidade principal deseja realizar.
- Recursos: um ou mais objetos de recurso da AWS em que ações ou operações são realizadas.
- Dados do recurso – Dados relacionados ao recurso que está sendo solicitado. Isso pode incluir informações como um nome da tabela do DynamoDB ou uma tag em uma instância do Amazon EC2.
- Dados do ambiente – Informações sobre o endereço IP, o agente de usuário, o status do SSL habilitado ou a hora do dia.

Essas informações são comparadas com as políticas aplicáveis para determinar se a solicitação deve ser permitida ou negada. Você pode organizar essas informações de propriedade usando o modelo Entidade principal, Ação, Recurso e Condição (PARC) para entender melhor como as políticas da AWS são avaliadas.

Entender o modelo PARC

O modelo PARC representa o contexto da solicitação com base nos quatro elementos em JSON do texto da política:

- Principal: a entidade que está fazendo a solicitação. Uma entidade principal do IAM representa um usuário humano ou uma workload programática que pode ser autenticada e depois autorizada a executar ações nas Contas da AWS.

- Action: a operação sendo realizada. Frequentemente, a ação será mapeada para uma ação da API.
- Resource: o recurso da AWS em que a ação está sendo realizada.
- Condition: restrições adicionais que devem ser atendidas para que a solicitação seja permitida.

O seguinte exemplo mostra como o modelo PARC poderia representar o contexto de uma solicitação:

```
Principal: AIDA123456789EXAMPLE
Action: s3:CreateBucket
Resource: arn:aws:s3:::amzn-s3-demo-bucket1
Context:
- aws:UserId=AIDA123456789EXAMPLE:BobsSession
- aws:PrincipalAccount=123456789012
- aws:PrincipalOrgId=o-example
- aws:PrincipalARN=arn:aws:iam::AIDA123456789EXAMPLE:role/HR
- aws:MultiFactorAuthPresent=true
- aws:CurrentTime=...
- aws:EpochTime=...
- aws:SourceIp=...
- aws:PrincipalTag/dept=123
- aws:PrincipalTag/project=blue
- aws:RequestTag/dept=123
```

Importância do contexto da solicitação

Entender o contexto da solicitação e como ele interage com a avaliação de políticas é essencial para:

- Solucionar problemas de acesso em geral
- Criar políticas eficazes e seguras
- Entender todo o escopo das permissões concedidas por uma política
- Prever o resultado das avaliações de políticas em diferentes cenários

Se visualizar o contexto da solicitação usando o modelo PARC, você poderá entender com mais facilidade como a AWS toma decisões de autorização e criar políticas mais eficazmente.

Como a AWS usa o contexto da solicitação

Ao avaliar políticas, a AWS compara as informações do contexto da solicitação com as informações especificadas em todas as políticas aplicáveis. Isso inclui as políticas baseadas em identidades, as políticas baseadas em recursos, os limites de permissões do IAM, as SCPs das organizações, as RCPs das organizações e as políticas de sessão.

Para cada tipo de política, a AWS usa o contexto da solicitação para verificar:

- Se a política se aplica à solicitação com base na entidade principal.
- Se a ação solicitada é permitida no recurso especificado.
- Se alguma das condições especificadas na política é atendida pelo contexto da solicitação.

A forma como o AWS avalia as políticas depende dos tipos de políticas que se aplicam ao contexto da solicitação. Esses tipos de políticas estão disponíveis para uso em uma única Conta da AWS. Para obter mais informações sobre esses tipos de política, consulte Políticas e permissões no AWS Identity and Access Management. Para saber como a AWS avalia políticas para acesso entre contas, consulte Lógica de avaliação de política entre contas.

- Políticas de controle de recursos (RCPs) do AWS Organizations: as RCPs do AWS Organizations especificam o número máximo de permissões disponíveis para recursos nas contas da sua organização ou unidade organizacional (UO). As RCPs se aplicam aos recursos nas contas de membros e afetam as permissões efetivas das entidades principais, incluindo o Usuário raiz da conta da AWS, independentemente de as entidades principais pertencerem à organização. As RCPs não se aplicam a recursos na conta de gerenciamento da organização e a chamadas feitas por perfis vinculados ao serviço. Se existir uma RCP, as permissões concedidas pelas políticas baseadas em identidades e baseadas em recursos às entidades principais nas contas de membros terão efeito apenas se a RCP permitir a ação.
- Políticas de controle de serviço (SCPs) do AWS Organizations: as SCPs do AWS Organizations especificam as permissões máximas disponíveis para entidades principais nas contas em uma organização ou unidade organizacional (UO). As SCPs se aplicam a entidades principais em contas-membro, incluindo cada Usuário raiz da conta da AWS. Se uma SCP estiver presente, as permissões concedidas por políticas baseadas em identidade e baseadas em recurso a entidades principais nas suas contas-membro somente estarão efetivas se a SCP permitir a ação. As únicas exceções são as entidades principais na conta de gerenciamento da organização e os perfis vinculados ao serviço.

- Políticas baseadas em recursos: as políticas baseadas em recursos concedem permissões para as entidades principais especificadas na política. As permissões definem o que a entidade principal pode fazer com o recurso ao qual a política está anexada.
- Limites de permissões: os limites de permissões são um atributo que define as permissões máximas que uma política baseada em identidades pode conceder a uma entidade do IAM (usuário ou perfil). Quando você definir um limite de permissões para uma entidade, a entidade poderá executar apenas as ações que sejam permitidas tanto pelas políticas baseadas em identidade quanto pelo seu limites de permissões. Em alguns casos, uma negação implícita em um limite de permissões pode limitar as permissões concedidas por uma política baseada em recursos. Para obter mais informações, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso.
- Políticas baseadas em identidade: as políticas baseadas em identidade são anexadas a uma identidade do IAM (usuário, grupo de usuários ou função) e concedem permissões a entidades do IAM (usuários e funções). Se somente políticas baseadas em identidade se aplicarem a uma solicitação, o AWS verificará todas essas políticas para pelo menos um `Allow`.
- Políticas de sessão: políticas de sessão são políticas que você passa como parâmetros ao criar de forma programática uma sessão temporária para um perfil ou sessão de usuário federado. Para criar uma sessão de função de forma programática, use uma das operações da API `AssumeRole*`. Quando você faz isso e passa políticas de sessão, as permissões de sessão resultantes são a interseção da política baseada em identidade do IAM e as políticas de sessão. Para criar uma sessão de usuário federado, use as chaves de acesso do usuário do IAM para chamar de forma programática a operação da API `GetFederationToken`. Para obter mais informações, consulte Políticas de sessão.

Lembre-se de que uma negação explícita em qualquer uma dessas políticas substitui a permissão.

> **Note**
>
> As políticas declarativas do AWS Organizations permitem que você declare e aplique centralmente a configuração desejada para um determinado AWS service (Serviço da AWS) em grande escala em uma organização. Como as políticas declarativas são aplicadas diretamente no nível do serviço, elas não afetam diretamente as solicitações de avaliação de políticas e não são incluídas no contexto da solicitação. Para obter mais informações, consulte Políticas declarativas no Guia do Usuário do AWS Organizations.

Exemplo de avaliação de políticas usando o modelo PARC

Para ilustrar como o contexto da solicitação interage com a avaliação da políticas, vamos considerar um exemplo de política:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:CreateBucket",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/dept": "123"
        }
      }
    }
  ]
}
```

Nesse exemplo, a política permitiria a ação `CreateBucket` apenas quando o contexto da solicitação incluísse um valor `aws:PrincipalTag/dept` de "123" e o recurso correspondesse ao nome de bucket `amzn-s3-demo-bucket1`. A tabela a seguir mostra como a AWS usa o contexto da solicitação para avaliar essa política e tomar decisões de autorização.

Política	Contexto da solicitação	Resultado da avaliação

`"StringEquals": { "aws:PrincipalTag/dept": "123" }`	`Principal: AIDA12345 6789EXAMPLE`
	`Action: s3:Create Bucket`
	`Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1`
	`Context:`
		correspondência

Política	Contexto da solicitação	Resultado da avaliação
	- aws:PrincipalTag/dept=123	
"StringEquals": { "aws:PrincipalTag/dept": "123" }	Principal: AIDA12345 6789EXAMPLE Action: s3: *DeleteBucket* Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1 Context: - aws:PrincipalTag/dept=123	Nenhuma correspondência
"StringEquals": { "aws:PrincipalTag/dept": "123" }	Principal: AIDA12345 6789EXAMPLE Action: s3:Create Bucket Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1 Context: - aws:PrincipalTag/dept= *321*	Nenhuma correspondência
"StringEquals": { "aws:PrincipalTag/dept": "123" }	Principal: AIDA12345 6789EXAMPLE Action: s3:Create Bucket Resource: arn:aws:s 3:::amzn-s3-demo-b ucket1 Context: Nenhum `aws:PrincipalTag` na solicitação.	Nenhuma correspondência

Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso

O código de imposição da AWS decide se a solicitação enviada a AWS deve ser permitida ou negada. A AWS avalia todas as políticas que são aplicáveis ao contexto da solicitação. A seguir há um resumo sobre a lógica de avaliação de políticas da AWS.

- Por padrão, todas as solicitações são implicitamente negadas, com exceção do Usuário raiz da conta da AWS, que tem acesso total.
- As solicitações devem ser explicitamente permitidas por uma política ou conjunto de políticas seguindo a lógica de avaliação abaixo para serem permitidas.
- Uma negação explícita se sobrepõe a uma permissão explícita.

A avaliação da política pode diferir dependendo se a solicitação está dentro de uma única conta ou é uma solicitação entre contas. Para obter detalhes de como uma decisão de avaliação de política é tomada para um perfil ou usuário do IAM em uma única conta, consulte Avaliação de políticas para solicitações em uma única conta. Para obter detalhes de como uma decisão de avaliação de política é tomada para solicitações entre contas, consulte Lógica de avaliação de política entre contas.

- Avaliação de negação: por padrão, todas as solicitações são negadas. Isso é chamado de negação implícita. O código de imposição do AWS avalia todas as políticas na conta que se aplicam à solicitação. Isso inclui SCPs e RCPs do AWS Organizations, políticas baseadas em recurso, políticas baseadas em identidade, limites de permissões do IAM e políticas de sessão. Em todas essas políticas, o código de imposição procura por uma declaração `Deny` aplicável à solicitação. Esse processo é chamado de negação explícita. Se o código de imposição encontrar uma única negação explícita aplicável, ele retornará uma decisão final de Negar. Se não houver uma negação explícita, a avaliação do código de imposição continuará.
- RCPs do AWS Organizations: o código de imposição avalia as políticas de controle de recursos (RCPs) do AWS Organizations que se aplicam à solicitação. As RCPs se aplicam aos recursos da conta à qual as RCPs estão vinculadas. Se o código de imposição não encontrar nenhuma instrução `Allow` aplicável nas RCPs, ele retornará uma decisão final de Negar. Observe que uma política gerenciada pela AWS chamada `RCPFullAWSAccess` é criada e anexada automaticamente a cada entidade em sua organização, incluindo a raiz, cada OU e Conta da AWS quando as RCPs estão habilitadas. A `RCPFullAWSAccess` não pode ser separada, então sempre haverá uma instrução `Allow`. Se não houver uma RCP ou se a RCP permitir a ação solicitada, a avaliação do código de imposição continuará.

- SCPs do AWS Organizations: o código de imposição avalia as políticas de controle de serviços (SCPs) do AWS Organizations que se aplicam à solicitação. As SCP aplicam-se às entidades principais da conta em que as SCP estão associadas. Se o código de imposição não encontrar nenhuma instrução `Allow` aplicável nas SCPs, ele retornará uma decisão final de Negar. Se não houver uma SCP ou se a SCP permitir a ação solicitada, a avaliação do código de imposição continuará.
- Políticas baseadas em recursos: na mesma conta, as políticas baseadas em recursos afetam a avaliação de política de maneira diferente, dependendo do tipo de entidade principal que está acessando o recurso e a entidade principal que é permitida na política baseada em recursos. Dependendo do tipo de entidade principal, um `Allow` em uma política baseada em recursos pode resultar em uma decisão final de `Allow`, mesmo se houver uma negação implícita em uma política baseada em identidade, limite de permissões ou política de sessão.

  Para a maioria dos recursos, você só precisa de um `Allow` explícito para a entidade principal em uma política baseada em identidade ou em recurso para conceder acesso. Políticas de confiança de perfil do IAM e políticas de chave do KMS são exceções a essa lógica, porque devem permitir explicitamente o acesso para entidades principais. As políticas baseadas em recursos para serviços diferentes do IAM e do AWS KMS também podem exigir uma declaração `Allow` explícita na mesma conta para conceder o acesso. Para obter mais informações, consulte a documentação do serviço específico com o qual está trabalhando.

  Para solicitações de avaliação de política em uma única conta, a lógica de política baseada em recurso difere de outros tipos de política se a entidade principal especificada for um usuário do IAM, um perfil do IAM ou uma entidade principal de sessão. As entidades principais de sessão incluem sessões de perfil do IAM ou entidades principais de usuário federado do AWS STS. Se uma política baseada em recursos conceder permissão diretamente ao usuário do IAM ou à entidade principal de sessão que está fazendo a solicitação, uma negação implícita em uma política baseada em identidade, um limite de permissões ou uma política de sessão não afetará a decisão final.

  - Função do IAM: as políticas baseadas em recursos que concedem permissões a um ARN de função do IAM são limitadas por uma negação implícita em um limite de permissões ou política de sessão. É possível especificar o ARN do perfil no elemento da entidade principal ou a chave de condição `aws:PrincipalArn`. Em ambos os casos, a entidade principal que faz a solicitação é a sessão de perfil do IAM.

    Limites de permissões ou políticas de sessão não limitam as permissões concedidas usando a chave de condição `aws:PrincipalArn` com um curinga (*) no elemento de entidade principal,

a menos que as políticas baseadas em identidade contenham uma negação explícita. Para obter mais informações, consulte Entidades de segurança de função do IAM.

Exemplo de ARN de função

```
arn:aws:iam::111122223333:role/examplerole
```

- Sessão de função do IAM: na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de sessão de função do IAM concedem permissões diretamente para a sessão de função assumida. As permissões concedidas diretamente a uma sessão não são limitadas por uma negação implícita em uma política baseada em identidade, um limite de permissões ou uma política de sessão. Quando você assume uma função e faz uma solicitação, a entidade principal que faz a solicitação é o ARN da sessão de função do IAM e não o ARN da função em si. Para obter mais informações, consulte Entidades principais da sessão de função.

  Exemplo de ARN de sessão de função

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- Usuário do IAM: na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de usuário do IAM (que não é uma sessão de usuário federado) não são limitadas por uma negação implícita em uma política baseada em identidade ou limite de permissões.

  Exemplo de ARN de usuário do IAM

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- Entidade principal de usuário federado do AWS STS: uma sessão de usuário federado é uma sessão criada por meio de um chamado a `GetFederationToken`. Quando um usuário federado faz uma solicitação, a entidade principal que faz a solicitação é o ARN do usuário federado e não o ARN do usuário do IAM que federou. Na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de usuário federado concedem permissões diretamente para a sessão. As permissões concedidas diretamente a uma sessão não são limitadas por uma negação implícita em uma política baseada em identidade, um limite de permissões ou uma política de sessão.

  No entanto, se uma política baseada em recursos conceder permissão ao ARN do usuário do IAM que se federou, as solicitações feitas pelo usuário federado durante a sessão serão limitadas por uma negação implícita em um limite de permissão ou política de sessão.

Exemplo de ARN de sessão de usuário federado

```
arn:aws:sts::111122223333:federated-user/exampleuser
```

- Políticas baseadas em identidade: o código de imposição verifica as políticas baseadas em identidade para a entidade principal. Para um usuário do IAM, isso inclui políticas de usuário e políticas de grupos aos quais o usuário pertence. Se não houver políticas baseadas em identidade ou instruções em políticas baseadas em identidade que permitam a ação solicitada, a solicitação será implicitamente negada e o código de imposição retornará uma decisão final de Negar. Se uma instrução em qualquer política aplicável baseada em identidade permitir a ação solicitada, a avaliação do código prosseguirá.
- Limites de permissões do IAM: o código de aplicação verifica se a entidade do IAM que é usada pela entidade principal tem um limite de permissões. Se a política que é usada para definir o limite de permissões não permitir a ação solicitada, a solicitação será implicitamente negada. O código de imposição retorna uma decisão final de Deny (Negação). Se não houver um limite de permissões ou se o limite de permissões permitir a ação solicitada, a avaliação do código prosseguirá.
- Políticas de sessão: o código de imposição confere se a entidade principal é uma entidade principal de sessão. As entidades principais de sessão incluem uma sessão de perfil do IAM ou uma sessão de usuário federado do AWS STS. Se a entidade principal não for uma entidade principal de sessão, o código de imposição retornará uma decisão final de Allow (Permitir).

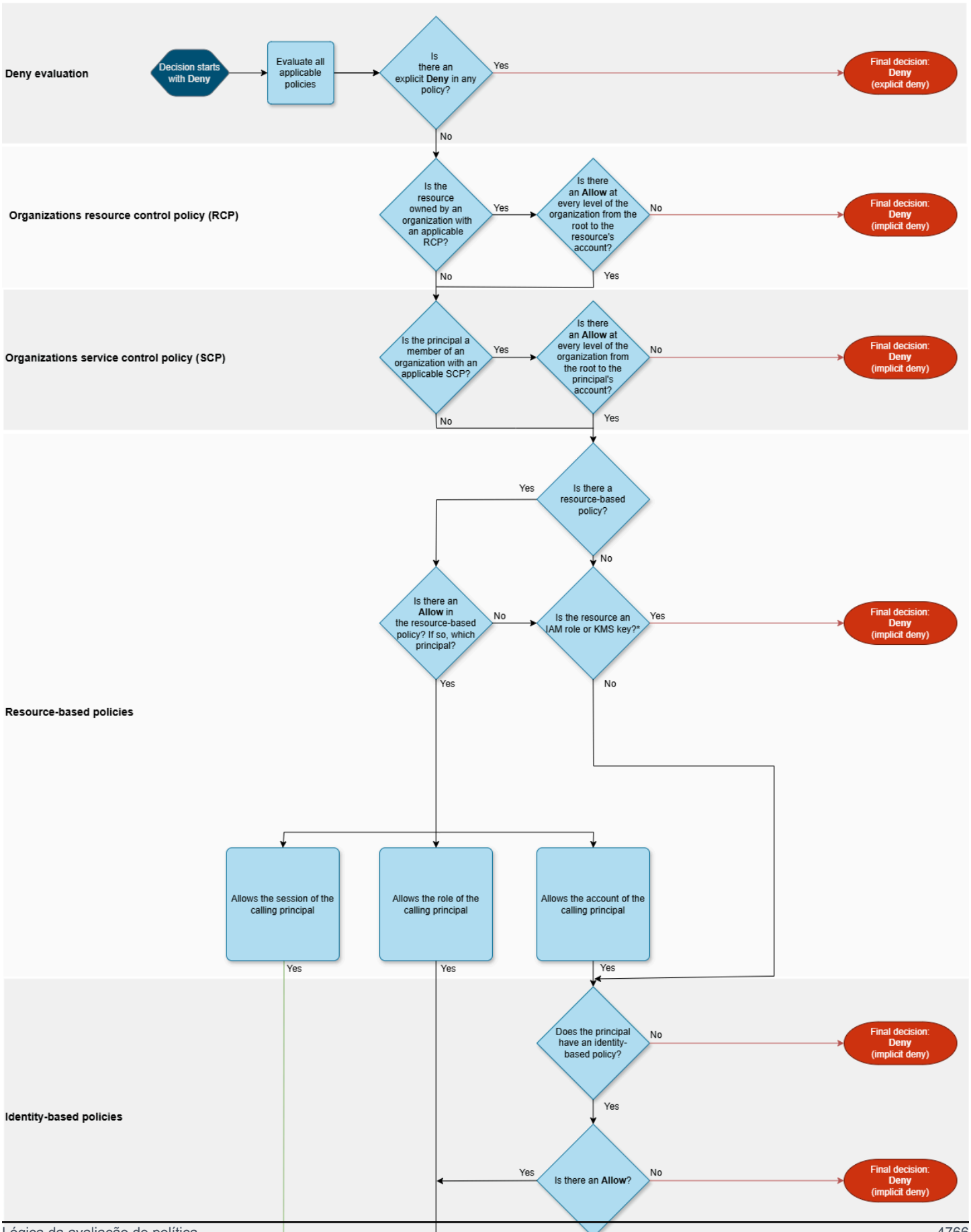
  Para entidades principais de sessão, o código de imposição confere se uma política de sessão foi transmitida na solicitação. É possível passar uma política de sessão enquanto usa a API AWS CLI ou AWS para obter credenciais temporárias para um perfil ou uma entidade principal de usuário federado do AWS STS. Se você não aprovou uma política de sessão, uma política de sessão padrão será criada, e o código de imposição retornará uma decisão final de Permitir.

  - Se houver uma política de sessão presente e ela não permitir a ação solicitada, a solicitação será implicitamente negada. O código de imposição retorna uma decisão final de Deny (Negação).
  - O código de imposição confere se a entidade principal é uma sessão de perfil. Se a entidade principal for uma sessão de perfil, a solicitação será Permitida. Caso contrário, a solicitação é implicitamente negada e o código de imposição retorna uma decisão final de Negar.
  - Se houver uma política de sessão presente e ela permitir a ação solicitada, o código de imposição retorna uma decisão final de Allow (Permitir).

Avaliação de políticas para solicitações em uma única conta

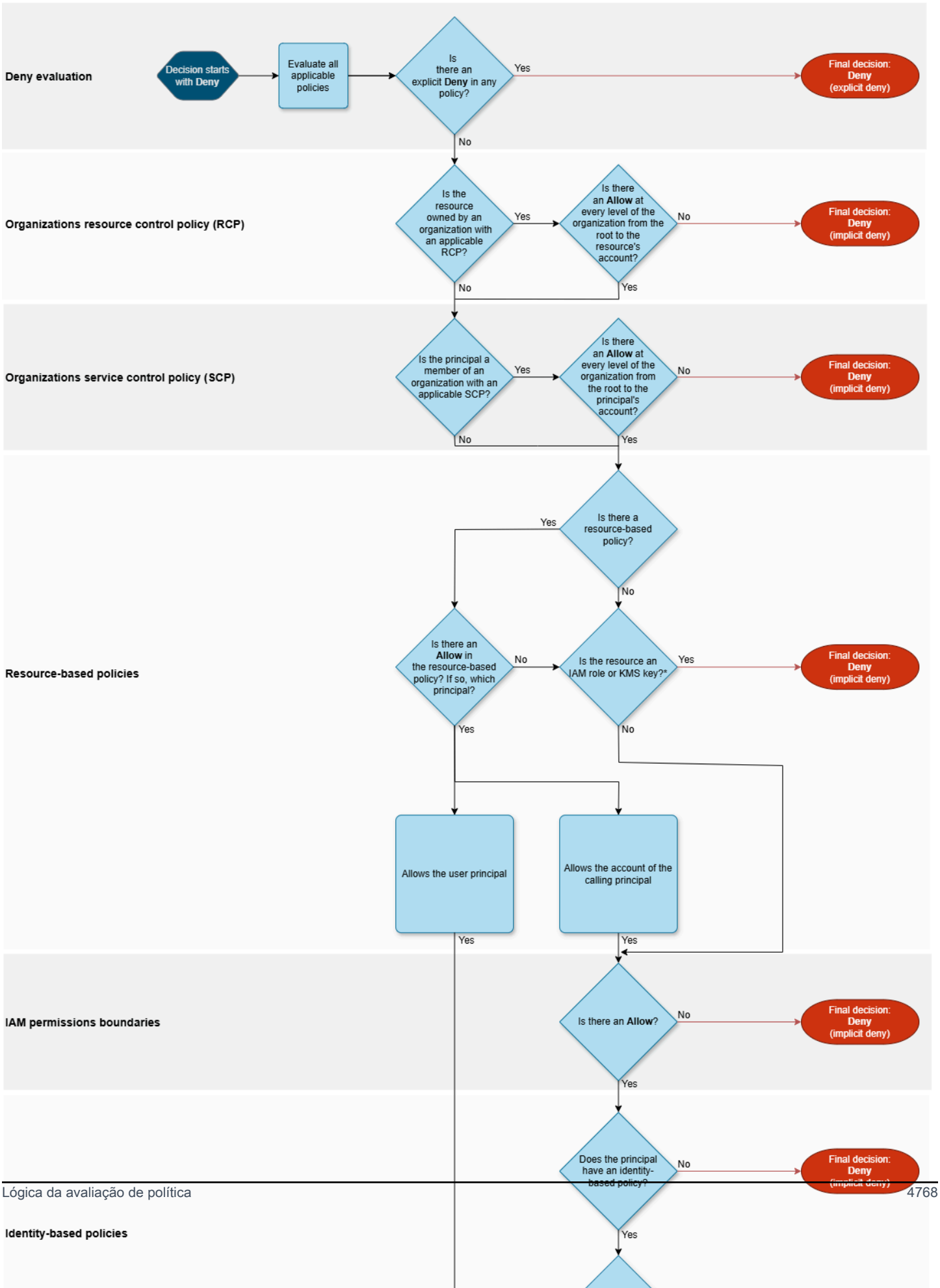
Avaliação de política para um perfil do IAM

O fluxograma a seguir fornece detalhes sobre como uma decisão de avaliação de política é tomada para um perfil do IAM em uma única conta.


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Is there a resource-based policy?
Yes
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the session of the calling principal
Allows the role of the calling principal
Allows the account of the calling principal
Yes
Yes
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes
Yes
Is there an Allow?
No
Final decision: Deny (implicit deny)
IAM permissions boundaries
Does the principal have a permissions
Yes
Is there an Allow?
No
Final decision: Deny

Avaliação de política para um usuário do IAM

O fluxograma a seguir fornece detalhes de como uma decisão de avaliação de política é tomada para um usuário do IAM em uma única conta.


Deny evaluation
Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny in any policy?
Yes
Final decision: Deny (explicit deny)
No
Organizations resource control policy (RCP)
Is the resource owned by an organization with an applicable RCP?
Yes
Is there an Allow at every level of the organization from the root to the resource's account?
No
Final decision: Deny (implicit deny)
No
Yes
Organizations service control policy (SCP)
Is the principal a member of an organization with an applicable SCP?
Yes
Is there an Allow at every level of the organization from the root to the principal's account?
No
Final decision: Deny (implicit deny)
No
Yes
Resource-based policies
Is there a resource-based policy?
Yes
No
Is there an Allow in the resource-based policy? If so, which principal?
No
Is the resource an IAM role or KMS key?*
Yes
Final decision: Deny (implicit deny)
Yes
No
Allows the user principal
Allows the account of the calling principal
Yes
IAM permissions boundaries
Is there an Allow?
No
Final decision: Deny (implicit deny)
Yes
Identity-based policies
Does the principal have an identity-based policy?
No
Final decision: Deny (implicit deny)
Yes

Exemplo de avaliação das políticas baseadas em identidade e políticas baseadas em recurso

Os tipos de políticas mais comuns são políticas baseadas em identidade e políticas baseadas em recurso. Quando o acesso a um recurso é solicitado, a AWS avalia todas as permissões concedidas pelas políticas para pelo menos uma permissão na mesma conta. Uma negação explícita em qualquer uma das políticas substitui a permissão.

> **Important**
>
> Se a política baseada em identidade ou a política baseada em recursos na mesma conta permitir a solicitação, mas a outra política não, a solicitação ainda será permitida.

Suponha que Carlos tem o nome de usuário `carlossalazar` e tente salvar um arquivo no bucket `amzn-s3-demo-bucket-carlossalazar-logs` do Amazon S3.

Suponha também que a política a seguir esteja anexada ao usuário do IAM `carlossalazar`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
```

```
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

A instrução `AllowS3ListRead` nessa política permite que Carlos visualize uma lista de todos os buckets da conta. A instrução `AllowS3Self` permite a Carlos acesso total ao bucket com o mesmo nome que seu nome de usuário. A instrução `DenyS3Logs` nega a Carlos acesso a qualquer bucket do S3 com `log` em seu nome.

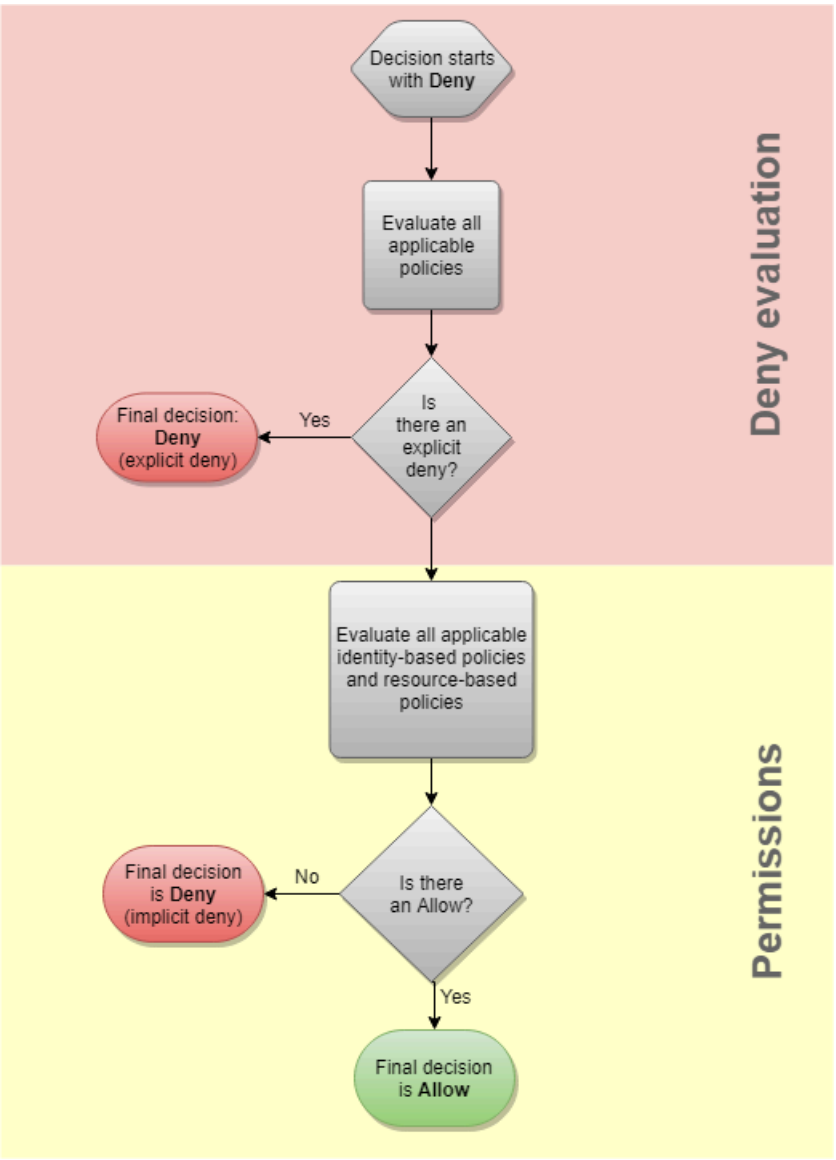
Além disso, a seguinte política baseada em recurso (chamada de política de bucket) está anexada ao bucket `amzn-s3-demo-bucket-carlossalazar`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar/*",
                "arn:aws:s3:::amzn-s3-demo-bucket-carlossalazar"
            ]
        }
    ]
}
```

Essa política especifica que apenas o usuário `carlossalazar` pode acessar o bucket `amzn-s3-demo-bucket-carlossalazar`.

Quando Carlos faz sua solicitação para salvar um arquivo no bucket `amzn-s3-demo-bucket-carlossalazar-logs`, a AWS determina quais políticas são aplicáveis à solicitação. Nesse caso, somente a política baseada em identidade e a política baseada em recurso são aplicáveis. Essas são duas políticas de permissões. Como nenhum limite de permissões se aplica, a lógica de avaliação é reduzida à lógica a seguir.



A AWS primeiro verifica se há uma instrução `Deny` aplicável ao contexto da solicitação. Ele encontra uma, pois a política baseada em identidade nega explicitamente a Carlos o acesso a qualquer bucket do S3 usado para log. O acesso é negado a Carlos.

Suponha que ele perceba seu erro e tente salvar o arquivo no bucket `amzn-s3-demo-bucket-carlossalazar`. A AWS verifica se há uma instrução `Deny` e não encontra uma. Em seguida, ela verifica a políticas de permissões. Tanto a política baseada em identidade quanto a política baseada em recursos permitem a solicitação. Portanto, a AWS permite a solicitação. Se qualquer uma delas tivesse negado explicitamente a instrução, a solicitação teria sido negada. Se um dos tipos de política permitir a solicitação e o outro não, a solicitação ainda será permitida.

Lógica de avaliação de política entre contas

É possível permitir que um principal em uma conta acesse os recursos em uma segunda conta. Isso é chamado de acesso entre contas. Quando você permite o acesso entre contas, a conta na qual o principal existe é chamada de conta confiável. A conta na qual o recurso existe é a conta de confiança.

Para permitir o acesso entre contas, anexe uma política baseada em recursos ao recurso que deseja compartilhar. É necessário anexar uma política baseada em identidade para a identidade que atua como entidade principal na solicitação. A política baseada em recursos na conta de confiança deve especificar o principal da conta confiável que terá acesso ao recurso. É possível especificar toda a conta ou seus usuários do IAM, entidades principais de usuários federados do AWS STS, perfis do IAM ou sessões de perfil assumido. Você também pode especificar um serviço da AWS como principal. Para obter mais informações, consulte Como especificar uma entidade principal.

A política baseada em identidade do principal deve permitir o acesso solicitado ao recurso no serviço de confiança. É possível fazer isso especificando o ARN do recurso.

No IAM, você pode anexar uma política baseada em recurso a uma função do IAM para permitir que entidades de segurança em outras contas assumam essa função. A política baseada em recursos da função é chamada de política de confiança da função. Depois de assumir essa função, os principais permitidos podem usar as credenciais temporárias resultantes para acessar vários recursos na conta. Esse acesso é definido na política de permissões baseadas na identidade da função. Para saber como a ação de permitir o acesso entre contas usando as funções difere da ação de permitir o acesso entre contas usando outras políticas baseadas em recursos, consulte Acesso a recursos entre contas no IAM.

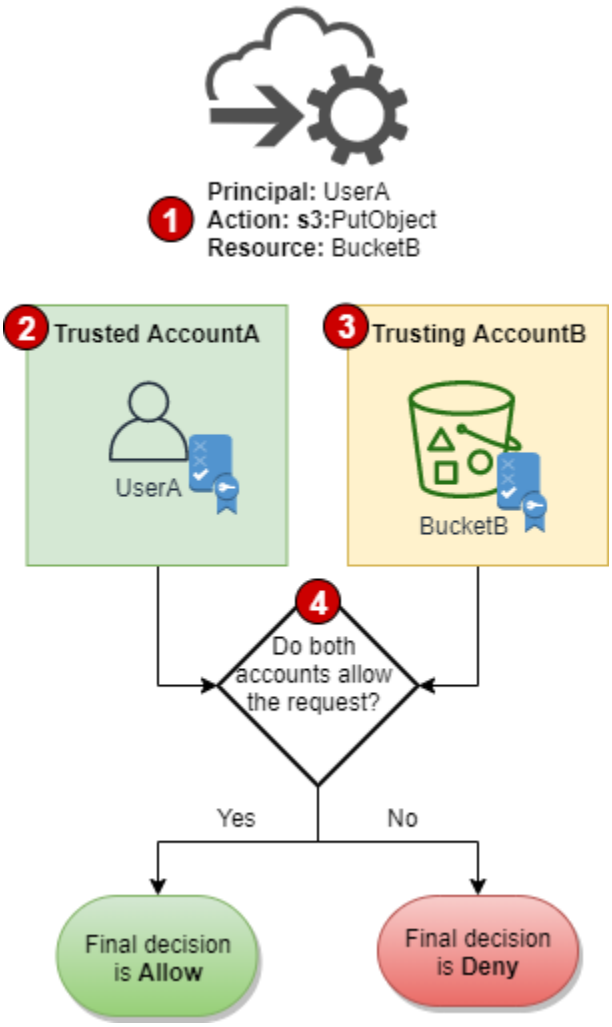
> **Important**
>
> Outros serviços podem afetar a lógica de avaliação da política. Por exemplo, o AWS Organizations oferece suporte a políticas de controle de serviço e políticas de controle de recursos que podem ser aplicadas a entidades principais e a recursos em uma ou mais

contas. O AWS Resource Access Manager oferece suporte a fragmentos de políticas que controlam quais ações as entidades principais têm permissão para executar nos recursos compartilhados com elas.

Determinar se uma solicitação entre contas é permitida

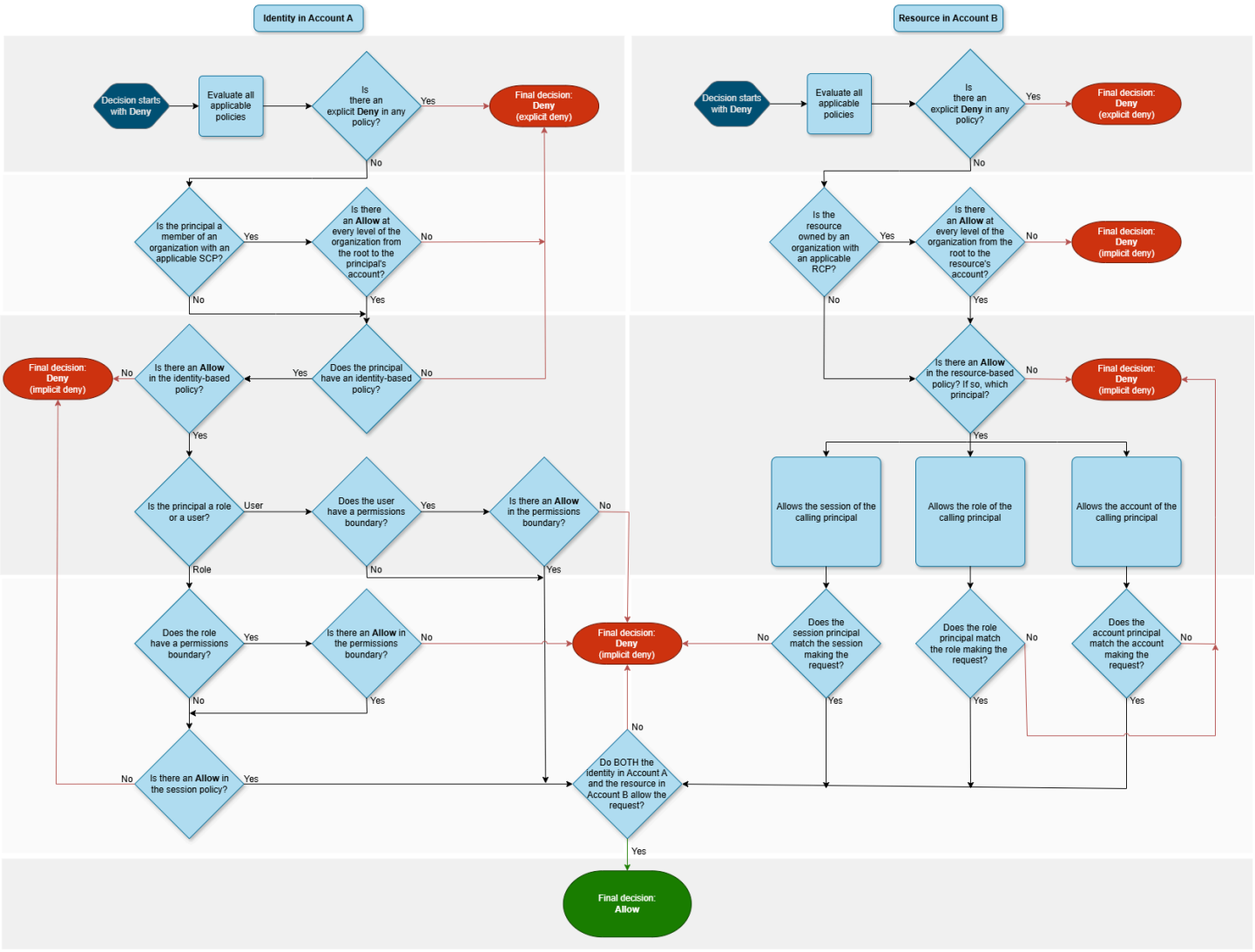
Para solicitações entre contas, o solicitante na `AccountA` confiável deve ter uma política baseada em identidade. Essa política deve permitir fazer uma solicitação para o recurso na de confiança `AccountB`. Além disso, a política baseada em recursos na `AccountB` deve permitir que o solicitante na `AccountA` acesse o recurso.

Ao fazer uma solicitação entre contas, a AWS executa duas avaliações. A AWS avalia a solicitação na conta de confiança e na conta confiável. Para obter mais informações sobre como uma solicitação é avaliada dentro de uma única conta, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso. A solicitação é permitida somente se ambas as avaliações retornarem uma decisão de `Allow`.



1. Quando um principal em uma conta fazer uma solicitação para acessar um recurso em outra conta, esta é uma solicitação entre contas.
2. O principal solicitante existe na conta confiável (`AccountA`). Quando a AWS avalia essa conta, ela verifica a política baseada em identidade e as políticas que podem limitar uma política baseada em identidade. Para obter mais informações, consulte Avaliação das políticas baseadas em identidade com limites de permissões.
3. O recurso solicitado existe na conta de confiança (`AccountB`). Quando a AWS avalia esta conta, ela verifica a política baseada em recursos que está em anexo ao recurso solicitado e todas as políticas que podem limitar uma política baseada em recursos. Para obter mais informações, consulte Avaliação das políticas baseadas em identidade com políticas baseadas em recurso.
4. AWSA permite a solicitação somente se ambas avaliações de política de conta permitirem a solicitação.

O fluxograma a seguir fornece uma ilustração mais detalhada de como uma decisão de avaliação de política é tomada para uma solicitação entre contas. Mais uma vez, a AWS permitirá a solicitação somente se ambas avaliações de política de conta permitirem a solicitação.



Exemplo de avaliação de política entre contas

O exemplo a seguir demonstra um cenário no qual um perfil em uma conta recebe permissões por meio de uma política baseada em recursos em uma segunda conta.

Suponha que Carlos seja um desenvolvedor com um perfil do IAM de nome `Demo` na conta 111111111111. Ele quer salvar um arquivo no bucket `amzn-s3-demo-bucket-production-logs` do Amazon S3 na conta 222222222222.

Suponha também que a política a seguir esteja anexada ao perfil do IAM `Demo`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

A instrução `AllowS3ListRead` nesta política permite que o Carlos visualize uma lista de todos os buckets no Amazon S3. A declaração `AllowS3ProductionObjectActions` permite que Carlos tenha acesso total a objetos no bucket `amzn-s3-demo-bucket-production`.

Além disso, a política baseada em recurso a seguir (chamada de política de bucket) está anexada ao bucket `amzn-s3-demo-bucket-production` na conta 222222222222.

JSON

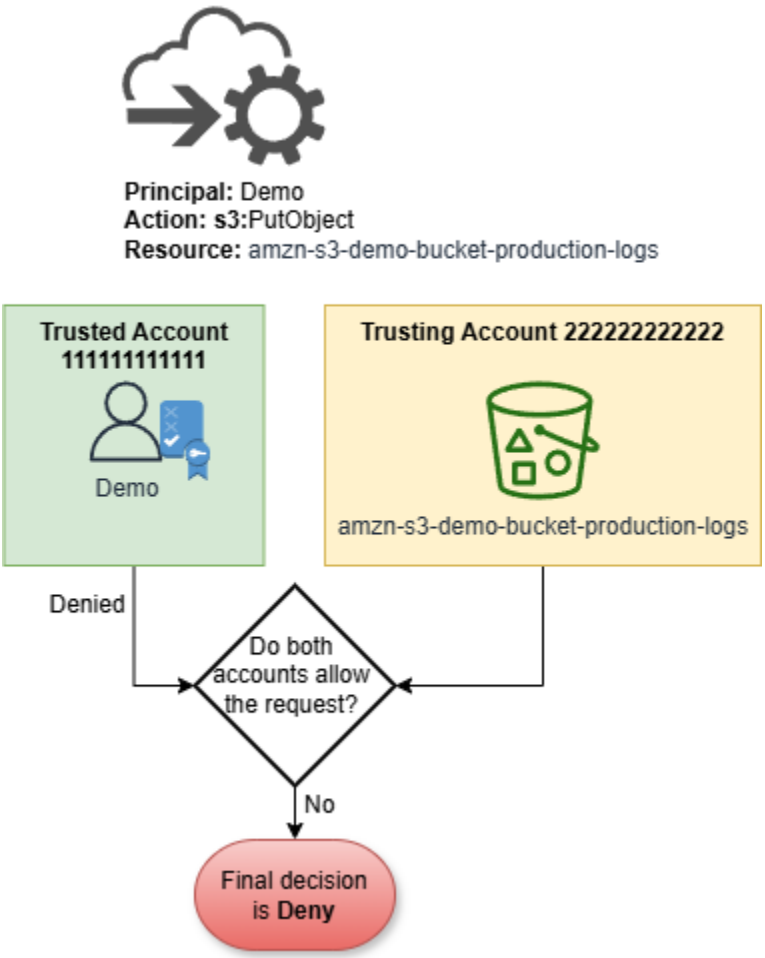
```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:role/Demo" },
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket-production/*"
        }
    ]
}
```

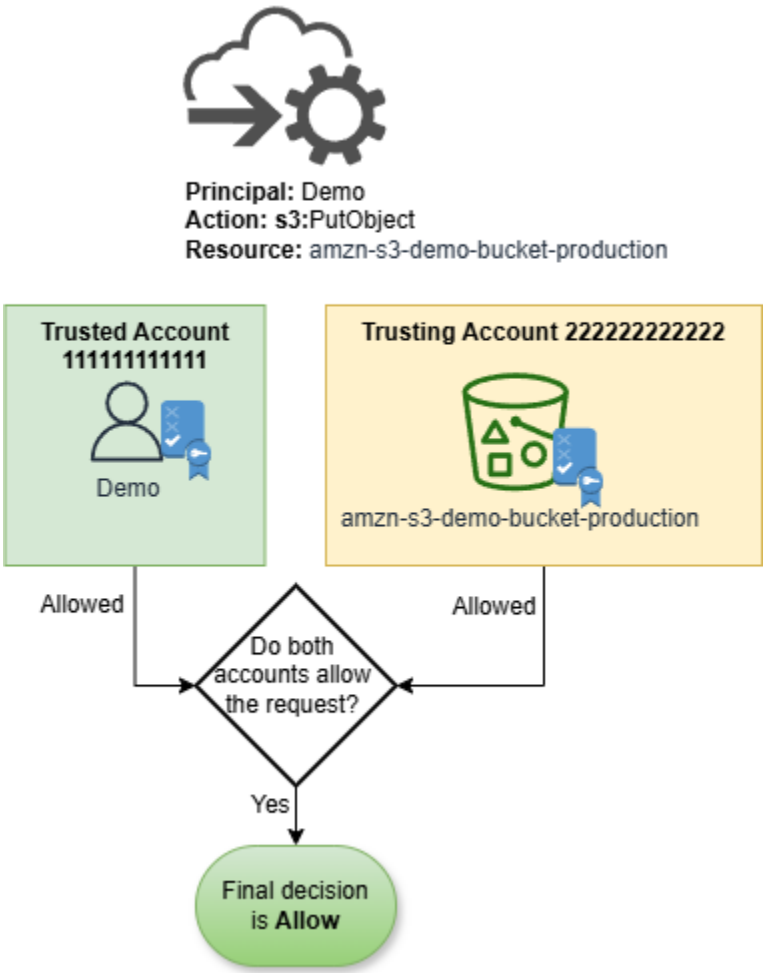
Essa política permite que o perfil `Demo` acesse objetos no bucket `amzn-s3-demo-bucket-production`. O perfil poderá criar e editar, mas não excluir os objetos no bucket. O perfil não conseguirá gerenciar o bucket propriamente dito.

Quando Carlos faz sua solicitação para salvar um arquivo no bucket `amzn-s3-demo-bucket-production-logs`, a AWS determina quais políticas são aplicáveis à solicitação. Nesse caso, a política baseada em identidade anexada ao perfil `Demo` é a única política aplicável à conta `111111111111`. Na conta `222222222222`, não há uma política baseada em recursos anexada ao bucket `amzn-s3-demo-bucket-production-logs`. Quando a AWS avalia a conta `111111111111`, ela retorna uma decisão de `Deny`. Isto ocorre porque a declaração `DenyS3Logs` na política baseada em identidade explicitamente nega o acesso a quaisquer buckets de log. Para obter mais informações sobre como uma solicitação é avaliada dentro de uma única conta, consulte Como a lógica do código de imposição da AWS avalia as solicitações para permitir ou negar acesso.

Como a solicitação é explicitamente negada dentro de uma das contas, a decisão final é negar a solicitação.



Suponha que o Carlos perceba o erro e tente salvar o arquivo no bucket `Production`. A AWS primeiro verifica a conta 111111111111 para determinar se a solicitação é permitida. Somente a política baseada em identidade se aplica e permite a solicitação. A AWS verificará a conta 222222222222. Somente a política baseada em recursos anexada ao bucket `Production` é aplicável, e permite a solicitação. Como ambas as contas permitem a solicitação, a decisão final é permitir a solicitação.



A diferença entre negações explícitas e implícitas

Uma solicitação resultará em uma negação explícita aplicável se uma política aplicável incluir uma instrução `Deny`. Se as políticas aplicáveis a uma solicitação incluírem uma instrução `Allow` e uma instrução `Deny`, a instrução `Deny` superará a instrução `Allow`. A solicitação será negada explicitamente.

Uma negação implícita ocorre quando não há uma instrução `Deny` aplicável, mas também nenhuma instrução `Allow` aplicável. Como o acesso da entidade principal do IAM é negado por padrão, ela deve ter permissão explícita para executar uma ação. Caso contrário, o acesso será implicitamente negado.

Ao projetar sua estratégia de autorização, você deve criar políticas com instruções `Allow` para permitir que suas entidades principais façam solicitações com êxito. No entanto, você pode escolher qualquer combinação de negações implícitas e explícitas.

Por exemplo, é possível criar a seguinte política que inclui ações permitidas, ações negadas implicitamente e ações negadas explicitamente. A declaração `AllowGetList` permite acesso do tipo somente leitura a ações do IAM que comecem com os prefixos `Get` e `List`. Todas as outras ações no IAM, como `iam:CreatePolicy`, são negadas implicitamente. A declaração `DenyReports` nega explicitamente o acesso a relatórios do IAM, negando acesso a ações que incluem o sufixo `Report`, como `iam:GetOrganizationsAccessReport`. Se alguém adicionar outra política a essa entidade principal para conceder acesso a relatórios do IAM, como `iam:GenerateCredentialReport`, solicitações relacionadas a relatórios ainda serão negadas por causa dessa negação explícita.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Gramática da linguagem das políticas de JSON do IAM

Esta página apresenta uma gramática formal para a linguagem usada para criar políticas de JSON no IAM. Apresentamos essa gramática para que você possa entender como construir e validar políticas.

Para obter exemplos de políticas, consulte os seguintes tópicos:

- Políticas e permissões no AWS Identity and Access Management
- Exemplos de políticas baseadas em identidade do IAM
- Políticas de exemplo para trabalhar no console do Amazon EC2 e Políticas de exemplo para trabalhar com a AWS CLI, a CLI do Amazon EC2ou um AWS SDK no Guia do usuário do Amazon EC2.
- Exemplos de políticas de bucket e Exemplos de política de usuário no Guia do usuário do Amazon Simple Storage Service.

Para obter exemplos de políticas usadas em outros produtos da AWS, consulte a documentação desses produtos.

Tópicos

- A linguagem das políticas e JSON
- Convenções usadas nesta gramática
- Gramática
- Notas sobre a gramática das políticas

A linguagem das políticas e JSON

As políticas são expressas em JSON. Quando você cria ou edita uma política JSON, o IAM pode executar a validação de políticas para ajudar você a criar uma política eficaz. O IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de política do IAM. Para saber mais sobre as verificações de política do IAM Access Analyzer e as recomendações práticas, consulte Validação de política do IAM Access Analyzer.

Neste documento, não fornecemos uma descrição completa dos elementos que constituem o JSON válido. No entanto, veja a seguir algumas regras básicas de JSON:

- Espaço em branco entre entidades individuais é permitido.
- Os valores são colocados entre aspas. Aspas são opcionais para valores numéricos e boolianos.
- Muitos elementos (por exemplo, `action_string_list` e `resource_string_list`) podem levar uma matriz JSON como um valor. As matrizes podem levar um ou mais valores. Se mais de

um valor for incluído, a matriz será inserida em colchetes ([e]) e delimitada por vírgula, como no exemplo a seguir:

```
"Action" : ["ec2:Describe*","ec2:List*"]
```

- Os tipos de dados JSON básicos (Booliano, número e string) são definidos na RFC 7159.

Convenções usadas nesta gramática

As seguintes convenções são usadas nesta gramática:

- Os caracteres a seguir são tokens JSON e são incluídos nas políticas:

  ```
  { } [ ] " , :
  ```

- Os caracteres a seguir são caracteres especiais na gramática e não são incluídos nas políticas:

  ```
  = < > ( ) |
  ```

- Se um elemento permitir vários valores, ele será indicado com valores repetidos, um delimitador de vírgula e reticências (...). Exemplos:

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Se vários valores forem permitidos, também será válido incluir apenas um valor. Para apenas um valor, a vírgula à direita deve ser omitida. Se o elemento levar uma matriz (marcada com [e]), mas apenas um valor for incluído, os colchetes serão opcionais. Exemplos:

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```

- Um ponto de interrogação (?) depois de um elemento indica que o elemento é opcional. Exemplo:

  ```
  <version_block?>
  ```

  No entanto, consulte as notas após a listagem de gramática para obter mais detalhes sobre elementos opcionais.

- Uma linha vertical (|) entre os elementos indica alternativas. Na gramática, parênteses definem o escopo das alternativas. Exemplo:

```
("Principal" | "NotPrincipal")
```

- Elementos que devem ser strings literais são inseridos entre aspas duplas ("). Exemplo:

```
<version_block> = "Version" : ("2008-10-17" | "2012-10-17" )
```

Para notas adicionais, consulte Notas sobre a gramática das políticas após a listagem de gramática.

Gramática

A lista a seguir descreve a gramática da linguagem das políticas. Para obter as convenções usadas na lista, consulte a seção anterior. Para obter informações adicionais, consulte as notas que se seguem.

> **Note**
>
> Essa gramática descreve as políticas marcadas com uma versão de `2008-10-17` e `2012-10-17`. Um elemento de política `Version` é diferente de uma versão de política. O elemento de política `Version` é usado em uma política e define a versão da linguagem da política. A versão da política, por outro lado, é criada quando você faz alterações em uma política gerenciada pelo cliente no IAM. A política alterada não substitui a política existente. Em vez disso, o IAM cria uma nova versão da política gerenciada. Para saber mais sobre o elemento de política `Version`, consulte Elementos de política JSON do IAM: Version. Para saber mais sobre as versões de política, consulte the section called “Versionamento de políticas do IAM”.

```
policy  = {
     <version_block?>,
     <id_block?>,
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17"        | "2012-10-17"       )

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
```

```
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | <action_string> | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Notas sobre a gramática das políticas

- Uma única política pode conter um conjunto de instruções.
- As políticas têm um tamanho máximo de 2048 caracteres e 10.240 caracteres, dependendo do elemento ao qual a política está anexada. Para obter mais informações, consulte IAM e cotas do AWS STS. Os cálculos de tamanho da política não incluem caracteres de espaço em branco.
- Elementos individuais não devem conter várias instâncias da mesma chave. Por exemplo, você não pode incluir o bloco `Effect` duas vezes na mesma instrução.

- Os blocos podem aparecer em qualquer ordem. Por exemplo, `version_block` pode seguir `id_block` em uma política. Da mesma forma, `effect_block`, `principal_block`, `action_block` podem aparecer em qualquer ordem dentro de uma instrução.
- O `id_block` é opcional em políticas baseadas em recursos. Não deve ser incluído em políticas baseadas em identidades.
- O elemento `principal_block` é necessário em políticas baseadas em recurso (por exemplo, em políticas de bucket do Amazon S3) e em políticas de confiança para funções do IAM. Não deve ser incluído em políticas baseadas em identidades.
- O elemento `principal_map` nas políticas de bucket do Amazon S3 pode incluir o ID `CanonicalUser`. A maioria das políticas baseadas em recursos não oferece suporte a esse mapeamento. Para saber mais sobre como usar o ID canônico de usuário em uma política de bucket, consulte Como especificar uma entidade principal em uma política no Guia do usuário do Amazon Simple Storage Service.
- Cada valor de string (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string` e a versão de string de `condition_value`) pode ter suas próprias restrições de tamanho mínimo e máximo, valores permitidos específicos ou o formato interno necessário.

Notas sobre valores de string

Esta seção fornece informações adicionais sobre valores de string que são usados em diferentes elementos em uma política.

`action_string`

Consiste em um namespace de serviço, dois pontos e o nome de uma ação. Os nomes de ação podem incluir curingas. Exemplos:

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"
```

```
"Action":[
  "s3:Get*",
  "s3:List*"
]
```

policy_id_string

Oferece uma maneira de incluir informações sobre a política como um todo. Alguns serviços, como o Amazon SQS e o Amazon SNS, usam o elemento `Id` de formas reservadas. A menos que isso seja proibido por um serviço individual, policy_id_string pode incluir espaços. Alguns serviços exigem que esse valor seja exclusivo em uma conta da AWS.

> **Note**
>
> O `id_block` é permitido em políticas baseadas em recursos, mas não em políticas baseadas em identidades.

Não há limite para o tamanho, embora essa string contribua para o comprimento geral da política, que é limitado.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

Oferece uma maneira de incluir informações sobre uma instrução individual. Para políticas do IAM, caracteres alfanuméricos básicos (A-Z, a-z, 0-9) são os únicos caracteres permitidos no valor `Sid`. Outros serviços da AWS que oferecem suporte a políticas de recursos podem ter outros requisitos para o valor `Sid`. Por exemplo, alguns serviços exigem esse valor para serem exclusivos em uma Conta da AWS, e alguns serviços permitem caracteres adicionais, como espaços no valor `Sid`.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

`principal_id_string`

Fornece uma maneira de especificar uma entidade de segurança usando o nome do recurso da Amazon (ARN) da Conta da AWS do usuário do IAM, do perfil do IAM, do usuário federado ou do usuário de perfil assumido. Para uma Conta da AWS, você também pode usar a forma curta `AWS:accountnumber` em vez de todo o ARN. Para todas as opções, incluindo serviços da AWS, funções assumidas etc., consulte Como especificar uma entidade principal.

Você pode usar * apenas para especificar "todos/anônimo". Não é possível usá-lo para especificar parte de um nome ou ARN.

`resource_string`

Na maioria dos casos, consiste em um nome de recurso da Amazon (ARN). Você pode usar os curingas (* e ?) na parte de recurso do ARN. Para obter mais informações sobre como usar o curinga em ARNs, consulte Usar curingas em caminhos.

> **Note**
>
> Quando você especifica um ARN incompleto (um com menos do que os seis campos padrão) em uma política baseada em identidade, a AWS preenche automaticamente o ARN adicionando caracteres curinga (*) a todos os campos ausentes. Por exemplo, especificar `arn:aws:sqs` é equivalente a `arn:aws:sqs:*:*:*`, o que concede acesso a todos os recursos do Amazon SQS em todas as regiões e contas. No entanto, as políticas de sessão passadas para as solicitações do AWS STS AssumeRole, AssumeRoleWithWebIdentity e AssumeRoleWithSAML não oferecem suporte a ARNs incompletos. Usar um ARN incompleto em uma política de sessão resultará em um erro `MalformedPolicyDocumentException`.

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::amzn-s3-demo-bucket/*"
```

`condition_type_string`

Identifica o tipo de condição que está sendo testado, como `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals`, etc. Para obter uma lista completa de tipos de condição, consulte Elementos de política JSON do IAM: operadores de condição.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Identifica a chave de condição cujo valor será testado para determinar se a condição foi atendida. A AWS define um conjunto de chaves de condição que estão disponíveis em todos os produtos da AWS, incluindo `aws:PrincipalType`, `aws:SecureTransport` e `aws:userid`.

Para obter uma lista de chaves de condição da AWS, consulte AWSChaves de contexto de condições globais da. Para chaves de condição que são específicas a um serviço, consulte a documentação do serviço, como o seguinte:

- Como especificar condições em uma política no Guia do usuário do Amazon Simple Storage Service
- Políticas do IAM para o Amazon EC2 no Guia do usuário do Amazon EC2.

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
```

```
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Identifica o valor de condition_key_string que determina se a condição foi atendida. Para obter uma lista completa de valores válidos para um tipo de condição, consulte Elementos de política JSON do IAM: operadores de condição.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
   "PostDateTime"
         ]
  }
}
```

AWSPolíticas gerenciadas pela para funções de trabalho

Recomendamos o uso de políticas que concedam privilégios mínimos ou conceder apenas as permissões necessárias para executar uma tarefa. A maneira mais segura de conceder privilégios mínimos é escrever uma política personalizada apenas com as permissões necessárias à sua equipe. Você deve criar um processo para permitir que sua equipe solicite mais permissões quando necessário. É necessário tempo e experiência para criar políticas gerenciadas pelo cliente do IAM que fornecem à sua equipe apenas as permissões de que precisam.

Para começar a adicionar permissões a suas identidades do IAM (usuários, grupos de usuários e perfis), você pode usar AWSPolíticas gerenciadas pela. As políticas gerenciadas pela AWS abrangem casos de uso comuns e estão disponíveis na sua Conta da AWS. As políticas gerenciadas AWS pela não concedem permissões de privilégio mínimo. Você deve considerar o risco de segurança de conceder às suas entidades de segurança mais permissões do que elas precisam para realizar um trabalho.

Você pode anexar políticas gerenciadas pela AWS, incluindo funções de trabalho, a qualquer identidade do IAM. Para alternar para permissões de privilégio mínimo, você pode executar o AWS Identity and Access Management and Access Analyzer para monitorar as entidades de segurança com políticas gerenciadas pela AWS. Depois de saber quais permissões elas estão usando, você pode escrever uma política personalizada ou gerar uma política apenas com as permissões necessárias para sua equipe. Isso é menos seguro, mas oferece mais flexibilidade à medida que você aprende como sua equipe está usando a AWS.

AWSPolíticas gerenciadas pela para funções de trabalho são projetadas para se alinhar de perto com funções de trabalho comuns no setor de TI. Você pode usar essas políticas para conceder as permissões necessárias para executar as tarefas esperadas de alguém com determinada função de trabalho. Essas políticas consolidam permissões para vários serviços em uma única política com a qual seja mais fácil trabalhar do que ter permissões espalhadas através de muitas políticas.

Use funções para combinar serviços

Algumas das políticas usam funções de serviço do IAM para ajudar você a aproveitar os recursos encontrados em outros produtos da AWS. Essas políticas concedem acesso a `iam:passrole`, o que permite que um usuário com a política passe uma função para um serviço da AWS. Essa função delega permissões do IAM para o produto da AWS executar ações em seu nome.

Você deve criar as funções de acordo com as suas necessidades. Por exemplo, a política de administrador de rede permite que um usuário com a política passe uma função chamada “flow-logs-vpc” para o serviço Amazon CloudWatch. O CloudWatch usa essa função para registrar em log e capturar o tráfego de IP para VPCs criadas pelo usuário.

Para seguir as melhores práticas de segurança, as políticas para funções de trabalho incluem filtros que limitam os nomes de funções válidas que podem ser transmitidas. Isso ajuda a evitar a concessão de permissões desnecessárias. Se seus usuários precisam das funções de serviço opcionais, você deve criar uma função que siga a convenção de nomenclatura especificada na política. Então, conceda permissões para a função. Ao concluir, o usuário pode configurar o serviço para usar a função, concedendo-lhe todas as permissões que a função oferece.

Nas seguintes seções, cada nome de política é um link para a página de detalhes da política no Console de gerenciamento da AWS. Lá, é possível consultar o documento de política e revisar as permissões que ela concede.

Função de trabalho de administrador

Nome da política gerenciada pela AWS: AdministratorAccess

Caso de uso: Este usuário tem acesso total e pode delegar permissões para todos os serviços e recursos na AWS.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: Esta política concede todas as ações para todos os serviços da AWS e para todos os recursos na conta. Para obter mais informações sobre a política gerenciada, consulte AdministratorAccess no Guia de referência de políticas gerenciadas pela AWS.

> **Note**
>
> Antes que um usuário ou uma função do IAM possa acessar o console do Gerenciamento de Faturamento e Custos da AWS com as permissões nesta política, você deve primeiro ativar o usuário e o acesso à função do IAM. Para fazer isso, siga as instruções em Grant access to the billing console para delegar acesso ao console de faturamento.

Função de trabalho de faturamento

Nome da política gerenciada pela AWS: Billing

Caso de uso: Este usuário precisa visualizar informações de faturamento, configurar e autorizar pagamentos. O usuário pode monitorar os custos acumulados para cada serviço da AWS.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: Esta política concede permissões completas para gerenciar faturamento, custos, meios de pagamento, orçamentos e relatórios. Para obter outros exemplos de políticas de gerenciamento de custos, consulte Exemplos de políticas do AWS Billing no Guia do usuário do Gerenciamento de Faturamento e Custos da AWS. Para obter mais informações sobre a política gerenciada, consulte Billing no Guia de referência de políticas gerenciadas pela AWS.

> **Note**
>
> Antes que um usuário ou uma função do IAM possa acessar o console do Gerenciamento de Faturamento e Custos da AWS com as permissões nesta política, você deve primeiro ativar o usuário e o acesso à função do IAM. Para fazer isso, siga as instruções em Grant access to the billing console para delegar acesso ao console de faturamento.

Função de trabalho de administrador de banco de dados

Nome da política gerenciada pela AWS: DatabaseAdministrator

Caso de uso: Este usuário define, configura e mantém bancos de dados na Nuvem AWS.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: Esta política concede permissões para criar, configurar e manter bancos de dados. Ela inclui acesso a serviços de banco de dados da AWS, como o Amazon DynamoDB, o Amazon Relational Database Service (RDS) e o Amazon Redshift. Visualize a política para obter a lista completa de serviços de banco de dados aos quais essa política oferece suporte. Para obter mais informações sobre a política gerenciada, consulte DatabaseAdministrator no Guia de referência de políticas gerenciadas pela AWS.

Essa política de função de trabalho oferece suporte à capacidade de passar funções para serviços da AWS. A política permite a ação `iam:PassRole` somente para as funções listadas na tabela a seguir. Para obter mais informações, consulte Criar funções e anexar políticas (console) mais adiante neste tópico.

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	Selecione esta política gerenciada pela AWS
Permitir que o usuário monitore bancos de dados do RDS	rds-monitoring-role	Função do Amazon RDS para monitoramento aprimorado	AmazonRDSEnhancedMonitoringRole

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	Selecione esta política gerenciada pela AWS
Permitir que AWS Lambda monitore seu banco de dados e acesse bancos de dados externos	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
Permitir que o Lambda carregue arquivos para o Amazon S3 e para clusters do Amazon Redshift com o DynamoDB	lambda_exec_role	AWS Lambda	Criar uma nova política gerenciada conforme definido no Blog de Big Data da AWS
Permitir que funções Lambda atuem como acionadores para suas tabelas do DynamoDB	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
Permitir que funções Lambda acessem o Amazon RDS em uma VPC	lambda-vpc-execution-role	Criar uma função com uma política de confiança conforme definido no Guia do desenvolvedor do AWS Lambda	AWSLambda VPCAccessExecutionRole

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	Selecione esta política gerenciada pela AWS
Permitir que o AWS Data Pipeline acesse seus recursos da AWS	DataPipelineDefaultRole	Criar uma função com uma política de confiança conforme definido no Guia do desenvolvedor do AWS Data Pipeline	A documentação do AWS Data Pipeline lista as permissões necessárias para este caso de uso. Consulte Perfis do IAM para o AWS Data Pipeline
Permitir que suas aplicações em execução em instâncias do Amazon EC2 acessem seus recursos da AWS	DataPipelineDefaultResourceRole	Criar uma função com uma política de confiança conforme definido no Guia do desenvolvedor do AWS Data Pipeline	AmazonEC2RoleforDataPipelineRole

Função de trabalho de cientista de dados

Nome da política gerenciada pela AWS: DataScientist

Caso de uso: Este usuário executa trabalhos e consultas do Hadoop. O usuário também acessa e analisa informações para data analytics e business intelligence.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: esta política concede permissões para criar, gerenciar e executar consultas em um cluster do Amazon EMR e executar data analytics com ferramentas como o Amazon QuickSight. A política inclui o acesso a serviços adicionais de cientistas de dados, como AWS Data Pipeline, Amazon EC2, Amazon Kinesis, Amazon Machine Learning e SageMaker. Visualize a política

para obter a lista completa de serviços de cientistas de dados aos quais essa política oferece suporte. Para obter mais informações sobre a política gerenciada, consulte DataScientist no Guia de referência de políticas gerenciadas pela AWS.

Essa política de função de trabalho oferece suporte à capacidade de passar funções para serviços da AWS. Uma instrução permite passar qualquer perfil para o SageMaker AI. Outra declaração permite a ação `iam:PassRole` somente para as funções listadas na tabela a seguir. Para obter mais informações, consulte Criar funções e anexar políticas (console) mais adiante neste tópico.

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	AWSPolítica gerenciada pela a selecionar
Permitir que instâncias do Amazon EC2 acessem serviços e recursos adequados para clusters	EMR-EC2_DefaultRole	Amazon EMR for EC2	AmazonElasticMapReduceforEC2Role
Permitir acesso ao Amazon EMR para acessar o serviço e recursos do Amazon EC2 para clusters	EMR_DefaultRole	Amazon EMR	AmazonEMRServicePolicy_v2
Permitir que o Kinesis Managed Service for Apache Flink acesse fontes de dados de streaming	kinesis-*	Criar uma função com uma política de confiança conforme definido no Blog de Big Data da AWS.	Consulte o Blog de Big Data da AWS, que descreve quatro opções possíveis, dependendo do caso de uso
Permitir que o AWS Data Pipeline acesse seus recursos da AWS	DataPipelineDefaultRole	Criar uma função com uma política de confiança conforme definido no Guia do desenvolvedor do AWS Data Pipeline	A documentação do AWS Data Pipeline lista as permissões necessárias para este caso de uso. Consulte Perfis do

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	AWSPolítica gerenciada pela a selecionar
			IAM para o AWS Data Pipeline
Permitir que suas aplicações em execução em instâncias do Amazon EC2 acessem seus recursos da AWS	DataPipelineDefaultResourceRole	Criar uma função com uma política de confiança conforme definido no Guia do desenvolvedor do AWS Data Pipeline	AmazonEC2RoleforDataPipelineRole

Função de trabalho de usuário desenvolvedor avançado

Nome da política gerenciada pela AWS: PowerUserAccess

Caso de uso: Este usuário executa tarefas de desenvolvimento de aplicativos e pode criar e configurar recursos e serviços compatíveis com o desenvolvimento consciente de aplicativos da AWS.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: a primeira declaração desta política utiliza o elemento `NotAction` para permitir todas as ações para todos os produtos da AWS e para todos os recursos, exceto para o AWS Identity and Access Management, o AWS Organizations e o AWS Gerenciamento de contas. A segunda instrução concede permissões do IAM para criar uma função vinculada ao serviço. Isso é necessário para alguns serviços que devem acessar recursos em outro serviço, como um bucket do Amazon S3.

Ela também concede permissões do AWS Organizations para visualizar informações sobre a organização do usuário, incluindo o e-mail da conta gerencial e as limitações da organização. Embora esta política limite o IAM e o AWS Organizations, ela permite que o usuário execute todas as ações do Centro de Identidade do IAM, se este recurso estiver habilitado. Ele também concede

permissões de gerenciamento de contas para visualizar quais regiões da AWS estão habilitadas ou desabilitadas para a conta.

Função de trabalho de administrador de rede

Nome da política gerenciada pela AWS: NetworkAdministrator

Caso de uso: Este usuário é responsável pela configuração e manutenção dos recursos de rede da AWS.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: esta política concede permissões para criar e manter recursos de rede no Auto Scaling, Amazon EC2, AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing, AWS Elastic Beanstalk, Amazon SNS, CloudWatch, CloudWatch Logs, Amazon S3, IAM, e Amazon Virtual Private Cloud. Para obter mais informações sobre a política gerenciada, consulte NetworkAdministrator no Guia de referência de políticas gerenciadas pela AWS.

Este perfil de trabalho requer a capacidade de passar perfis para serviços da AWS. A política concede `iam:GetRole` e `iam:PassRole` apenas para as funções listadas na seguinte tabela. Para obter mais informações, consulte Criar funções e anexar políticas (console) mais adiante neste tópico.

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	AWSPolítica gerenciada pela a selecionar
Permite que o Amazon VPC crie e gerencie logs no CloudWatch Logs em nome do usuário para monitorar o tráfego de IP que entra e sai de sua VPC	flow-logs-*	Criar uma função com uma política de confiança conforme definido no Guia do usuário do Amazon VPC	Este caso de uso não tem uma política gerenciad a pela AWS existente, mas a documentação lista as permissõe s necessárias. Consulte Guia do

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	AWSPolítica gerenciada pela a selecionar
			usuário do Amazon VPC.

Acesso somente leitura.

Nome da política gerenciada pela AWS: ReadOnlyAccess

Caso de uso: este usuário requer acesso somente leitura a todos os recursos em uma Conta da AWS.

> **Important**
>
> Esse usuário também terá acesso para leitura de dados nos serviços de armazenamento, como buckets do Amazon S3 e tabelas do Amazon DynamoDB.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política:: esta política concede permissões para listar, obter, descrever e, de outra forma, visualizar recursos e seus atributos. Ela não inclui funções de mudança como criar ou excluir. Esta política inclui acesso somente leitura a produtos da AWS relacionados à segurança, como AWS Identity and Access Management e Gerenciamento de Faturamento e Custos da AWS.

Visualize a política para obter a lista completa de serviços e ações a que esta política oferece suporte. Para obter mais informações sobre a política gerenciada, consulte ReadOnlyAccess no AWS Managed Policy Reference Guide. Se você precisar de uma política similar que não conceda acesso para leitura de dados nos serviços de armazenamento, consulte Função de trabalho de usuário somente para visualização.

Acesso total às ações do serviço MCP

Nome da política gerenciada pela AWS: AWSMcpServiceActionsFullAccess

Caso de uso: este usuário requer acesso aos serviços da AWS usando servidores MCP da AWS. Esta política não concede acesso às ações executadas por um serviço MCP a outros serviços da AWS.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: esta política concede permissões para chamar qualquer ação do serviço MCP da AWS. Você pode usar quando não precisar especificar permissões por serviço MCP da AWS. Ele não concede permissões para ações executadas pelo serviço MCP a outros serviços da AWS. Essas permissões sempre devem ser concedidas separadamente de forma complementar às ações do serviço MCP. Para obter mais informações sobre a política gerenciada, consulte AWSMcpServiceActionsFullAccess no Guia de referência de políticas gerenciadas pela AWS.

Função de trabalho do auditor de segurança

Nome da política gerenciada pela AWS: SecurityAudit

Caso de uso: Este usuário monitora contas quanto à conformidade com os requisitos de segurança. Este usuário pode acessar registros e eventos para investigar potenciais violações de segurança ou possíveis atividades maliciosas.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: Esta política concede permissões para visualizar dados de configuração para muitos serviços da AWS e para revisar seus registros. Para obter mais informações sobre a política gerenciada, consulte SecurityAudit no Guia de referência de políticas gerenciadas pela AWS.

Função de trabalho de usuário do suporte

Nome da política gerenciada da AWS: AWSSupportAccess

Caso de uso: Este usuário entra em contato com o Suporte da AWS cria casos de suporte e visualiza o status de casos existentes.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: esta política concede permissões para criar e atualizar casos de Suporte. Para obter mais informações sobre a política gerenciada, consulte AWSSupportAccess no Guia de referência de políticas gerenciadas da AWS.

Função de trabalho de administrador de sistema

Nome da política gerenciada pela AWS: SystemAdministrator

Caso de uso: Este usuário configura e mantém recursos para as operações de desenvolvimento.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política:: esta política concede permissões para criar e manter recursos em uma grande variedade de serviços da AWS, incluindo o AWS CloudTrail, Amazon CloudWatch, AWS CodeCommit, AWS CodeDeploy, AWS Config, AWS Directory Service, Amazon EC2, AWS Identity and Access Management, AWS Key Management Service, AWS Lambda, Amazon RDS, Route 53, Amazon S3, Amazon SES, Amazon SQS, AWS Trusted Advisor e Amazon VPC. Para obter mais informações sobre a política gerenciada, consulte SystemAdministrator no Guia de referência de políticas gerenciadas pela AWS.

Este perfil de trabalho requer a capacidade de passar perfis para serviços da AWS. A política concede `iam:GetRole` e `iam:PassRole` apenas para as funções listadas na seguinte tabela. Para obter mais informações, consulte Criar funções e anexar políticas (console) mais adiante neste tópico. Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Caso de uso	Nome do perfil (* é um curinga)	Tipo do perfil de serviço a selecionar	AWSPolítica gerenciada pela a selecionar
Permitir que aplicações em execução em instâncias do EC2 em um cluster do Amazon ECS acessem o Amazon ECS	ecr-sysadmin-*	Função do Amazon EC2 para o EC2 Container Service	AmazonEC2 ContainerServicefo rEC2Role
Permitir que um usuário monitore bancos de dados	rds-monitoring-role	Função do Amazon RDS para monitoramento aprimorado	AmazonRDS EnhancedM onitoringRole
Permitir que aplicativos em execução em instâncias do EC2 acessem recursos da AWS.	ec2-sysadmin-*	Amazon EC2	Amostra de política para perfil que concede acesso a um bucket do S3, conforme apresentado no Guia do usuário do Amazon EC2; personalizar conforme necessári o
Permitir que o Lambda leia fluxos do DynamoDB e grave no CloudWatch Logs	lambda-sysadmin-*	AWS Lambda	AWSLambda DynamoDBE xecutionRole

Função de trabalho de usuário somente para visualização

Nome da política gerenciada pela AWS: ViewOnlyAccess

Caso de uso: este usuário pode visualizar uma lista de recursos e metadados básicos da AWS na conta nos serviços. O usuário não pode ler o conteúdo do recurso ou metadados além da cota e informações da lista para os recursos.

Atualizações da política: a AWS mantém e atualiza esta política. Para obter um histórico de alterações para esta política, visualize a política no console do IAM e escolha a guia Policy versions (Versões da política). Para obter mais informações sobre atualizações de políticas de funções de trabalho, consulte Atualizações nas políticas gerenciadas pela AWS para funções de trabalho.

Descrição da política: esta política concede a `List*`, `Describe*`, `Get*`, `View*` e `Lookup*` acesso a recursos para serviços da AWS. Para ver quais ações essa política inclui para cada serviço, consulte ViewOnlyAccess. Para obter mais informações sobre a política gerenciada, consulte ViewOnlyAccess no Guia de referência de políticas gerenciadas pela AWS.

Atualizações nas políticas gerenciadas pela AWS para funções de trabalho

Todas essas políticas são mantidas pela AWS e atualizadas para incluir suporte a novos serviços e novos recursos à medida que são adicionados pela AWS. Essas políticas não podem ser modificadas pelos clientes. Você pode fazer uma cópia da política e, em seguida, modificar a cópia, mas essa cópia não é atualizada automaticamente à medida que a AWS introduz novos serviços e operações de API.

Para uma política de função de trabalho, você pode visualizar o histórico de versões e a hora e a data de cada atualização no console do IAM. Para fazer isso, use os links desta página para visualizar os detalhes da política. Em seguida, escolha a guia Policy versions (Versões da política) para visualizar as versões. Esta página mostra as últimas 25 versões de uma política. Para visualizar todas as versões de uma política, chame o comando get-policy-version da AWS CLI ou a operação GetPolicyVersion da API.

> **Note**
>
> Você pode ter até cinco versões de uma política gerenciada pelo cliente, mas a AWS mantém o histórico de versões completo das políticas gerenciadas pela AWS.

Criar funções e anexar políticas (console)

Várias das políticas previamente listadas concedem a capacidade de configurar serviços da AWS com funções que permitem que esses serviços executem operações em seu nome. As políticas de

função de trabalho especificam nomes de função exatos que você deve usar ou, no mínimo, incluem um prefixo que especifica a primeira parte do nome que pode ser usado. Para criar uma dessas funções, execute as etapas no procedimento a seguir.

Para criar uma função para um AWS service (Serviço da AWS) (console do IAM)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
3. Em Tipo de entidade confiável, escolha AWS service (Serviço da AWS).
4. Para Serviço ou caso de uso, escolha um serviço e, em seguida, escolha o caso de uso. Casos de uso são definidos pelo serviço para incluir a política de confiança exigida pelo serviço.
5. Escolha Próximo.
6. As opções para Políticas de permissões dependem do caso de uso selecionado.
   - Se o serviço definir as permissões para o perfil, não será possível selecionar políticas de permissões.
   - Selecione em um conjunto limitado de políticas de permissões.
   - Selecione entre todas as políticas de permissões.
   - Não selecione nenhuma política de permissão; crie políticas após a criação do perfil e, em seguida, anexe as políticas ao perfil.
7. (Opcional) Defina um limite de permissões. Esse é um atributo avançado que está disponível para perfis de serviço, mas não para perfis vinculados ao serviço.
   a. Abra a seção Definir limite de permissões e escolha Usar um limite de permissões para controlar o número máximo de permissões do perfil.

      O IAM inclui uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta.
   b. Selecione a política a ser usada para o limite de permissões.
8. Escolha Próximo.
9. Para Nome do perfil, as opções dependem do serviço:
   - Se o serviço definir o nome do perfil, não será possível editar esse nome.
   - Se o serviço definir um prefixo para o nome do perfil, você poderá inserir um sufixo opcional.
   - Se o serviço definir o nome do perfil, você poderá atribuir um nome ao perfil.

> **Important**
>
> Quando nomear um perfil, observe o seguinte:
>
> - Os nomes do perfil devem ser exclusivos em sua Conta da AWS e não podem ser diferenciados caso a caso.
>
> Por exemplo, não crie dois perfis denominados **`PRODROLE`** e **`prodrole`**. Quando usado em uma política ou como parte de um ARN, o nome de perfil diferencia maiúsculas de minúsculas. No entanto, quando exibido para os clientes no console, como durante o processo de login, o nome de perfil diferencia maiúsculas de minúsculas.
> - Não é possível editar o nome do perfil depois de criá-lo porque outras entidades podem referenciar o perfil.

10. (Opcional) Em Descrição, insira uma descrição para o perfil.
11. (Opcional) Para editar os casos de uso e as permissões do perfil, escolha Editar nas seções Etapa 1: selecionar entidades confiáveis ou Etapa 2: adicionar permissões.
12. (Opcional) Para ajudar a identificar, organizar ou pesquisar o perfil, adicione tags como pares de chave-valor. Para obter mais informações sobre o uso de tags no IAM, consulte Tags para recursos do AWS Identity and Access Management no Guia do usuário do IAM.
13. Reveja a função e escolha Criar função.

Exemplo 1: Configuração de um usuário como um administrador de banco de dados (console)

Este exemplo mostra as etapas necessárias para configurar Alice, uma usuária do IAM, como administrador de banco de dados. É possível usar as informações na primeira linha da tabela na seção e permitir que o usuário habilite o monitoramento do Amazon RDS. Anexe a política DatabaseAdministrator ao usuário do IAM de Alice para que ela possa gerenciar os serviços de banco de dados da Amazon. Essa política também permite que Alice passe ao serviço Amazon RDS um perfil denominado `rds-monitoring-role` que permite que o serviço monitore os bancos de dados do Amazon RDS em nome dela.

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Selecione Políticas, digite **`database`** na caixa de pesquisa e pressione enter.

3. Marque o botão de seleção da política DatabaseAdministrator, escolha Ações e depois Anexar.
4. Na lista de usuários, selecione Alice e depois escolha Anexar política. Alice agora pode administrar bancos de dados da AWS. No entanto, para permitir que Alice monitore esses bancos de dados, você deve configurar a função de serviço.
5. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
6. Escolha o tipo de perfil Serviço da AWS e escolha Amazon RDS.
7. Escolha o caso de uso Amazon RDS Role for Enhanced Monitoring (Função do Amazon RDS para monitoramento avançado).
8. O Amazon RDS define as permissões para a função. Escolha Próximo: Revisar para continuar.
9. O nome da função deve ser um dos nomes especificados pela política DatabaseAdministrator que Alice agora possui. Um deles é **`rds-monitoring-role`**. Insira esse nome em Role name (Nome do perfil).
10. (Opcional) Em Descrição do perfil, insira uma descrição para o novo perfil.
11. Após revisar os detalhes, selecione Create role (Criar função).
12. Alice agora pode habilitar o RDS Enhanced Monitoring (Monitoramento avançado do RDS) na seção Monitoring (Monitoramento) do console do Amazon RDS. Por exemplo, ela poderia fazer isso ao criar uma instância de banco de dados ou uma réplica de leitura, ou ao modificar uma instância de banco de dados. Ela deve inserir o nome do perfil que criou (rds-monitoring-role) na caixa Monitoring Role (Perfil de monitoramento) quando definir Enable Enhanced Monitoring (Habilitar monitoramento avançado) como Yes (Sim).

Exemplo 2: Configuração de um usuário como um administrador de rede (console)

Este exemplo mostra as etapas necessárias para configurar Jorge, um usuário do IAM, como administrador da rede. O exemplo usa as informações da tabela nessa seção para permitir que Jorge monitore o tráfego de IP entrando e saindo de uma VPC. Permite também que Jorge capture essas informações nos logs do CloudWatch Logs. Anexe a política NetworkAdministrator ao usuário do IAM de Jorge para que ele possa configurar os recursos de rede da AWS. Essa política também permitirá que Jorge passe um perfil cujo nome começa com `flow-logs*` para o Amazon EC2 quando você criar um log de fluxo. Nesse cenário, ao contrário do Exemplo 1, não há um tipo de função de serviço predefinido; portanto, você deve realizar algumas etapas de forma diferente.

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação, selecione Políticas e insira **network** na caixa de pesquisa, depois pressione enter.
3. Marque o botão de seleção ao lado da política NetworkAdministrator, escolha Ações e depois escolha Anexar.
4. Na lista de usuários, marque a caixa de seleção ao lado de Jorge e selecione Anexar política. Jorge agora pode administrar os recursos de rede da AWS. No entanto, para habilitar o monitoramento de tráfego de IP em sua VPC, você deve configurar a função de serviço.
5. Como a função de serviço que você precisa criar não tem uma política gerenciada predefinida, você deve primeiro criá-la. No painel de navegação, selecione Políticas e, em seguida, Criar política.
6. Na seção Editor de políticas, escolha a opção JSON e copie o texto do documento da política JSON a seguir. Cole este texto na caixa de texto do JSON.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

7. Resolva os avisos de segurança, as mensagens erros ou os avisos gerais gerados durante a validação de política, e depois escolha Próximo.

> **Note**
>
> É possível alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar no editor Visual, o IAM poderá

reestruturar a política a fim de otimizá-la para o editor visual. Para obter mais informações, consulte Reestruturação da política.

8. Na página Revisar e criar, digite **`vpc-flow-logs-policy-for-service-role`** para o nome da política. Revise Permissões definidas nessa política para ver as permissões concedidas pela política e depois escolha Criar política para salvar seu trabalho.

   A nova política aparece na lista de políticas gerenciadas e está pronta para ser anexada.
9. No painel de navegação do console do IAM, escolha Perfis e, em seguida, Criar perfil.
10. Escolha o tipo de perfil Serviço da AWS e escolha Amazon EC2.
11. Selecione o caso de uso Amazon EC2.
12. Na página Anexar políticas de permissão, selecione a política que você criou anteriormente, vpc-flow-logs-policy-for-service-role e, em seguida, selecione Próxima: Revisar.
13. O nome da função deve ser permitido pela política NetworkAdministrator que Jorge agora tem. Qualquer nome que comece com `flow-logs-` é permitido. Neste exemplo, insira **`flow-logs-for-jorge`** em Role name (Nome do perfil).
14. (Opcional) Em Descrição do perfil, insira uma descrição para o novo perfil.
15. Após revisar os detalhes, selecione Create role (Criar função).
16. Agora você pode configurar a política de confiança necessária para este cenário. Na página Perfis, selecione o perfil flow-logs-for-jorge (o nome, não a caixa de seleção). Na página de detalhes para a sua nova função, selecione a guia Relações de confiança e, em seguida, selecione Editar relação de confiança.
17. Altere o "Serviço" para ler da seguinte forma, substituindo a entrada por `ec2.amazonaws.com`:

    ```
            "Service": "vpc-flow-logs.amazonaws.com"
    ```
18. Jorge agora pode criar logs de fluxo para uma VPC ou sub-rede no console do Amazon EC2. Quando você criar o log de fluxos, especifique o perfil flow-logs-for-jorge. Essa função tem as permissões para criar o log e gravar dados nele.

AWSChaves de contexto de condições globais da

Quando um principal faz uma solicitação à AWS, a AWS reúne as informações da solicitação em um contexto de solicitação. É possível usar o elemento `Condition` de uma política JSON para comparar chaves no contexto da solicitação com os valores de chave especificados em sua política.

As informações da solicitação são fornecidas por fontes diferentes, incluindo a entidade principal que faz a solicitação, o recurso contra o qual a solicitação é feita e os metadados sobre a solicitação em si.

As chaves de condição globais podem ser usadas em todos os serviços da AWS. Embora essas chaves de condição possam ser usadas em todas as políticas, a chave não está disponível em todos os contextos de solicitação. Por exemplo, a chave de condição `aws:SourceAccount` só está disponível quando a chamada para seu recurso é feita diretamente por um responsável pela entidade principal do serviço da AWS. Para saber mais sobre as circunstâncias em que uma chave global é incluída no contexto da solicitação, consulte as informações de Disponibilidade de cada chave de condição global.

Alguns serviços individuais criam suas próprias chaves de condição que estão disponíveis no contexto da solicitação para outros serviços. As chaves de condição entre serviços são um tipo de chave de condição global que inclui um prefixo correspondente ao nome do serviço, como `ec2:` ou `lambda:`, mas estão disponíveis em outros serviços.

As chaves de condição específicas do serviço são definidas para uso com um serviço da AWS individual. Por exemplo, o Amazon S3 permite que você escreva uma política com a chave de condição `s3:VersionId` para limitar o acesso a uma versão específica de um objeto do Amazon S3. Essa chave de condição é exclusiva do serviço, o que significa que ela só funciona com solicitações ao serviço Amazon S3. Para chaves de condição específicas de serviços, consulte Ações, recursos e chaves de condição para serviços da AWS e escolha o serviço cujas chaves deseja visualizar.

> **Note**
>
> Se você usar chaves de condição disponíveis apenas em algumas circunstâncias, será possível usar as versões IfExists dos operadores de condição. Se as chaves de condição estiverem ausentes no contexto de uma solicitação, poderá ocorrer falha na avaliação da política. Por exemplo, use o seguinte bloqueio de condição com operadores `...IfExists` para saber quando uma solicitação é proveniente de um determinado intervalo de IP ou de uma determinada VPC. Se uma ou ambas as chaves não estiverem incluídas no contexto da solicitação, a condição ainda retornará `true`. Os valores serão verificados somente se a chave especificada estiver incluída no contexto da solicitação. Para obter mais informações sobre como uma política é avaliada quando uma chave não está presente para outros operadores, consulte Operadores de condição.

```
"Condition": {
    "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
    "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
}
```

Important

Para comparar sua condição com um contexto de solicitação com vários valores de chave, você deve usar os operadores de conjunto `ForAllValues` ou `ForAnyValue`. Use operadores de conjunto somente com chaves de condições de vários valores. Não use operadores de conjuntos com chaves de condição de valor único. Para obter mais informações, consulte Operadores de conjunto para chaves de contexto de vários valores.

Propriedades da entidade principal	Propriedades de uma sessão de perfil	Propriedades da rede	Propriedades do recurso	Propriedades da solicitação
aws:PrincipalArn	aws:AssumedRoot	aws:SourceIp	aws:ResourceAccount	aws:CalledVia
aws:PrincipalAccount	aws:FederatedProvider	aws:SourceVpc	aws:ResourceOrgID	aws:CalledViaFirst
aws:PrincipalOrgPaths	aws:TokenIssueTime	aws:SourceVpcArn	aws:ResourceOrgPaths	aws:CalledViaLast
aws:PrincipalOrgID	aws:SignInSessionArn	aws:SourceVpce	aws:ResourceTag/ *tag-key*	aws:CalledViaAWSMCP
aws:PrincipalTag/ *tag-key*	aws:MultiFactorAuthAge	aws:VpceAccount		aws:ViaAWSService
		aws:VpceOrgID		aws:ViaAWSMCPService

Propriedades da entidade principal	Propriedades de uma sessão de perfil	Propriedades da rede	Propriedades do recurso	Propriedades da solicitação
aws:PrincipalIsAWSService aws:PrincipalServiceName aws:PrincipalServiceNamesList aws:PrincipalType aws:userid aws:username	aws:MultiFactorAuthPresent aws:ChatbotSourceArn aws:Ec2InstanceSourceVpc aws:Ec2InstanceSourcePrivateIPv4 aws:SourceIdentity aws:ViaCustomerDomain ec2:RoleDelivery ec2:SourceInstanceArn glue:RoleAssumedBy glue:CredentialIss	aws:VpceOrgPaths aws:VpcSourceIp		aws:CurrentTime aws:EpochTime aws:referer aws:RequestedRegion aws:RequestTag/ *tag-key* aws:TagKeys aws:SecureTransport aws:SourceAccount aws:SourceArn aws:SourceOrgID aws:SourceOrgPaths aws:UserAgent

Propriedades da entidade principal	Propriedades de uma sessão de perfil	Propriedades da rede	Propriedades do recurso	Propriedades da solicitação
	uingServi ce codebuild :BuildArn codebuild :ProjectA rn lambda:So urceFunct ionArn ssm:Sourc eInstance Arn identitys tore:User Id			aws:IsMcp ServiceAc tion

Chaves de condição sensíveis

As seguintes chaves de condição são consideradas confidenciais. Não existe nenhum caso válido de uso de curingas nessas chaves de condição, mesmo com uma substring do valor da chave com um curinga. Isso ocorre porque o curinga pode fazer a correspondência da chave de condição com qualquer valor, o que pode representar um risco de segurança.

- aws:PrincipalAccount
- aws:PrincipalOrgID
- aws:ResourceAccount
- aws:ResourceOrgID

- aws:SourceAccount
- aws:SourceOrgID
- aws:SourceVpc
- aws:SourceVpce
- aws:VpceAccount
- aws:VpceOrgID

Propriedades da entidade principal

Use as chaves de condição a seguir para comparar detalhes sobre a entidade principal que está fazendo a solicitação com as propriedades da entidade principal especificada na política. Para obter uma lista das entidades principais que podem fazer solicitações, consulte Como especificar uma entidade principal.

aws:PrincipalArn

Use essa chave para comparar o nome do recurso da Amazon (ARN) do principal que fez a solicitação com o ARN especificado na política. Para funções do IAM, o contexto da solicitação retorna o ARN da função, não o ARN do usuário que assumiu a função.

- Disponibilidade: essa chave será incluída no contexto da solicitação para todas as solicitações assinadas. As solicitações anônimas não incluem essa chave. É possível especificar os seguintes tipos de entidades principais nesta chave de condição:
  - IAM role (Perfil do IAM)
  - IAM user (Usuário do IAM)
  - AWS STSEntidade principal de usuário federado do
  - Conta da AWSUsuário-raiz
- Tipo de dados: ARN

  A AWS recomenda utilizar operadores ARN em vez de operadores string ao comparar ARNs.
- Tipo de valor: valor único
- Valores de exemplo: a lista a seguir mostra o valor do contexto de solicitação retornado para diferentes tipos de entidades principais que você pode especificar na chave de condição `aws:PrincipalArn`:

- Perfil do IAM: o contexto da solicitação contém o seguinte valor para a chave de condição `aws:PrincipalArn`. Não especifique o ARN da sessão de perfil assumida como um valor para essa chave de condição. Para obter mais informações sobre a entidade principal da sessão de perfil assumida, consulte Entidades principais da sessão de função.

```
arn:aws:iam::123456789012:role/role-name
```

- Usuário do IAM: o contexto da solicitação contém o seguinte valor para a chave de condição `aws:PrincipalArn`.

```
arn:aws:iam::123456789012:user/user-name
```

- Entidades principais de usuário federado do AWS STS: o contexto da solicitação contém o valor a seguir para a chave de condição `aws:PrincipalArn`.

```
arn:aws:sts::123456789012:federated-user/user-name
```

- Usuário-raiz da Conta da AWS: o contexto da solicitação contém o valor a seguir para a chave de condição `aws:PrincipalArn`. Quando você especifica o ARN do usuário-raiz como o valor da chave de condição `aws:PrincipalArn`, ele limita as permissões apenas para o usuário-raiz da Conta da AWS. Isso é diferente de especificar o ARN do usuário-raiz no elemento principal de uma política baseada em recursos, o qual delega autoridade à Conta da AWS. Para obter mais informações sobre como especificar o ARN do usuário-raiz no elemento principal de uma política baseada em recursos, consulte Conta da AWSEntidades principais da.

```
arn:aws:iam::123456789012:root
```

É possível especificar o ARN do usuário-raiz como um valor para o `aws:PrincipalArn` da chave de condição nas políticas de controle de serviço (SCPs) do AWS Organizations. As SCPs são um tipo de política da organização usada para gerenciar permissões em sua organização e afetam apenas contas-membro na organização. Uma SCP restringe as permissões para usuários e funções do IAM em contas-membro, incluindo o usuário-raiz da conta-membro. Para obter mais informações sobre o efeito das SCPs nas permissões, consulte Efeitos de SCP sobre permissões no Guia do usuário do AWS Organizations.

aws:PrincipalAccount

Use essa chave para comparar a conta à qual o principal solicitante pertence com o identificador de conta especificado na política. Para solicitações anônimas, o contexto da solicitação retorna `anonymous`.

- Availability (Disponibilidade): essa chave é incluída no contexto da solicitação para todas as solicitações, incluindo as anônimas.
- Tipo de dados: String
- Tipo de valor: valor único

No exemplo a seguir, o acesso é negado, exceto para entidades principais com o número de conta `123456789012`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:resource"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws:PrincipalOrgPaths

Use esta chave para comparar o caminho AWS Organizations para o principal que está fazendo a solicitação para o caminho na política. Essa entidade principal pode ser um usuário do IAM, um perfil do IAM, uma entidade principal de usuário federado do AWS STS ou Usuário raiz da conta da AWS. Em uma política, essa chave de condição garante que o solicitante seja um membro da conta na raiz da organização especificada ou unidades organizacionais (OUs) em AWS Organizations. Um caminho AWS Organizations é uma representação de texto da estrutura de uma entidade do AWS Organizations. Para obter mais informações sobre como usar e entender caminhos, consulte Compreender o caminho da entidade do AWS Organizations.

- Disponibilidade: essa chave é incluída no contexto da solicitação somente se a entidade principal for membro de uma organização. As solicitações anônimas não incluem essa chave.
- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos

> **Note**
>
> Os IDs de organização são globalmente exclusivos, mas os IDs da UO e da raiz são exclusivos somente dentro de uma organização. Isso significa que duas organizações não compartilham o mesmo ID de organização. No entanto, outra organização pode ter uma UO ou raiz com o mesmo ID que o seu. Recomendamos sempre incluir o ID da organização ao especificar uma UO ou raiz.

Por exemplo, a condição a seguir retorna `true` para entidades de segurança em contas anexadas diretamente à UO `ou-ab12-22222222`, mas não em suas UOs filhas.

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

A condição a seguir retorna `true` para entidades de segurança em uma conta que está anexada diretamente à UO ou a qualquer uma de suas UOs filhas. Ao incluir um caractere curinga, é necessário usar o operador de condição `StringLike`.

```
"Condition" : { "ForAnyValue:StringLike" : {
```

```
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

A condição a seguir retorna `true` para entidades de segurança em uma conta que está anexada diretamente a qualquer uma das UOs filhas, mas não diretamente à UO mãe. A condição anterior é para a UO ou para as crianças. A condição a seguir é apenas para filhos (e quaisquer filhos desses filhos).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

A condição a seguir permite acesso a todos os principais da organização `o-a1b2c3d4e5`, independentemente de sua UO pai.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` é uma chave de condição de vários valores. Chaves de valores múltiplos podem ter vários valores no contexto da solicitação. Quando você usa vários valores com o operador de condição `ForAnyValue`, o caminho do principal deve corresponder a um dos caminhos listados na política. Para obter mais informações sobre chaves de condição de vários valores, consulte Operadores de conjunto para chaves de contexto de vários valores.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws:PrincipalOrgID

Use essa chave para comparar o identificador da organização no AWS Organizations ao qual o principal solicitante pertence com o identificador especificado na política.

- Disponibilidade: essa chave é incluída no contexto da solicitação somente se a entidade principal for membro de uma organização. As solicitações anônimas não incluem essa chave.
- Tipo de dados: String
- Tipo de valor: valor único

Essa chave global fornece uma alternativa para listar todos os IDs de conta para todas as contas da AWS em uma organização. É possível usar essa chave de condição para simplificar a especificação do elemento `Principal` em uma política baseada no recurso. É possível especificar o ID da organização no elemento de condição. Ao adicionar e remover contas, as políticas que incluem a chave `aws:PrincipalOrgID` incluem automaticamente as contas corretas e não exigem a atualização manual.

Por exemplo, a política de bucket do Amazon S3 a seguir permite que membros de qualquer conta na organização `o-xxxxxxxxxxx` adicionem um objeto ao bucket `amzn-s3-demo-bucket`.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
   "Condition": {"StringEquals":
     {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
   }
 }
}
```

> **Note**
>
> Essa condição global também se aplica a conta de gerenciamento de uma organização da AWS. Essa política impede que todas as entidades principais fora da organização especificada acessem o bucket do Amazon S3. Isso inclui quaisquer serviços da AWS que interagem com seus recursos internos, como o AWS CloudTrail enviando dados de log para

seus buckets do Amazon S3. Para saber mais sobre como conceder acesso a serviços da AWS com segurança, consulte aws:PrincipalIsAWSService.

Para obter informações sobre o AWS Organizations, consulte O que é AWS Organizations? no Guia do usuário do AWS Organizations.

aws:PrincipalTag/tag-key

Use essa chave para comparar a tag anexada ao principal que está fazendo a solicitação com a tag especificada na política. Se o principal tiver mais de uma tag anexada, o contexto da solicitação incluirá uma chave `aws:PrincipalTag` para cada chave de tag anexada.

- Disponibilidade: essa chave será incluída no contexto da solicitação se a entidade principal estiver usando um usuário do IAM com etiquetas anexadas. Ela será incluída para uma entidade principal usando um perfil do IAM com tags anexadas ou etiquetas de sessão. As solicitações anônimas não incluem essa chave.
- Tipo de dados: String
- Tipo de valor: valor único

É possível adicionar atributos personalizados a um usuário ou uma função na forma de um par de chave/valor. Para obter mais informações sobre etiquetas do IAM, consulte Tags para recursos do AWS Identity and Access Management. É possível usar `aws:PrincipalTag` para controlar o acesso de entidades principais da AWS.

Este exemplo mostra como você pode criar uma política baseada em identidade que permita que usuários com a tag **`department=hr`** gerenciem usuários, grupos ou perfis do IAM. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "iam:Get*",
        "iam:List*",
        "iam:Generate*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws:PrincipalIsAWSService

Use esta chave para verificar se a chamada para o seu recurso está sendo feita diretamente por uma entidade principal do produto da AWS. Por exemplo, o AWS CloudTrail usa a entidade principal `cloudtrail.amazonaws.com` do serviço para gravar logs no bucket do Amazon S3. A chave de contexto de solicitação é definida como verdadeira quando um serviço usa uma entidade principal do serviço para executar uma ação direta em seus recursos. A chave de contexto é definida como false (falsa) se o serviço usar as credenciais de uma entidade principal do IAM para fazer uma solicitação em nome da entidade principal. Também é definida como falsa se o serviço usar um perfil de serviço ou perfil vinculado ao serviço para fazer uma chamada em nome da entidade principal.

- Disponibilidade do: essa chave está presente no contexto da solicitação para todas as solicitações de API assinadas que usam credenciais da AWS. As solicitações anônimas não incluem essa chave.
- Tipo de dados: Booleano
- Tipo de valor: valor único

É possível usar essa chave de condição para limitar o acesso às suas identidades confiáveis e locais de rede esperados, ao mesmo tempo que concede acesso com segurança aos produtos da AWS.

No exemplo de política de bucket do Amazon S3 a seguir, o acesso ao bucket é restrito, a menos que a solicitação seja originada de `vpc-111bbb22` ou seja de uma entidade principal de serviço, como o CloudTrail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWSLogs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

No vídeo a seguir, saiba mais sobre como você pode usar a chave de condição `aws:PrincipalIsAWSService` em uma política.

Conceda com segurança acesso a seus usuários autorizados, locais de rede esperados e serviços da AWS ao mesmo tempo.

aws:PrincipalServiceName

Use esta chave para comparar o nome da entidade principal de serviço na política com a entidade principal de serviço que está fazendo solicitações aos seus recursos. É possível usar essa chave para verificar se essa chamada é feita por uma entidade principal de serviço específica. Quando uma entidade principal do serviço faz uma solicitação direta ao seu recurso, a chave `aws:PrincipalServiceName` contém o nome da entidade principal do serviço. Por exemplo, o nome da entidade principal do serviço AWS CloudTrail é `cloudtrail.amazonaws.com`.

- Disponibilidade: essa chave está presente na solicitação quando a chamada é feita por uma entidade principal de produto da AWS. Esta chave não está presente em nenhuma outra situação, incluindo a seguinte:
  - Se o serviço usa um perfil de serviço ou perfil vinculado ao serviço para fazer uma chamada em nome da entidade principal de segurança.
  - Se o serviço usar as credenciais de uma entidade principal do IAM para fazer uma solicitação em nome da entidade principal.
  - Se a chamada for feita diretamente por uma entidade principal do IAM.
  - Se a chamada for feita por um solicitante anônimo.
- Tipo de dados: String
- Tipo de valor: valor único

É possível usar essa chave de condição para limitar o acesso às suas identidades confiáveis e locais de rede esperados, ao mesmo tempo que concede acesso com segurança a um produto da AWS.

No exemplo de política de bucket do Amazon S3 a seguir, o acesso ao bucket é restrito, a menos que a solicitação seja originada de `vpc-111bbb22` ou seja de uma entidade principal de serviço, como o CloudTrail.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ExpectedNetworkServicePrincipal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/AWSLogs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
```

```
}
```

aws:PrincipalServiceNamesList

Essa chave fornece uma lista de todos os nomes de entidade principal de serviço que pertencem ao serviço. Esta é uma chave de condição avançada. É possível usá-la para restringir o acesso do serviço ao seu recurso somente de uma região específica. Alguns serviços podem criar entidades principais de serviço regionais para indicar uma instância específica do serviço dentro de uma região específica. É possível limitar o acesso a um recurso para uma instância específica do serviço. Quando uma entidade principal do serviço faz uma solicitação direta ao seu recurso, a chave `aws:PrincipalServiceNamesList` contém uma lista não ordenada de todos os nomes das entidades de segurança do serviço associadas à instância regional do serviço.

- Disponibilidade: essa chave está presente na solicitação quando a chamada é feita por uma entidade principal de produto da AWS. Esta chave não está presente em nenhuma outra situação, incluindo a seguinte:
  - Se o serviço usa um perfil de serviço ou perfil vinculado ao serviço para fazer uma chamada em nome da entidade principal de segurança.
  - Se o serviço usar as credenciais de uma entidade principal do IAM para fazer uma solicitação em nome da entidade principal.
  - Se a chamada for feita diretamente por uma entidade principal do IAM.
  - Se a chamada for feita por um solicitante anônimo.
- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos

`aws:PrincipalServiceNamesList` é uma chave de condição de vários valores. Chaves de valores múltiplos podem ter vários valores no contexto da solicitação. Você deve usar os operadores de conjunto `ForAnyValue` ou `ForAllValues` com operadores de condição de string para essa chave. Para obter mais informações sobre chaves de condição de vários valores, consulte Operadores de conjunto para chaves de contexto de vários valores.

aws:PrincipalType

Use essa chave para comparar o tipo de principal que está fazendo a solicitação com o tipo de principal especificado na política. Para obter mais informações, consulte Como especificar uma

entidade principal. Para exemplos específicos de valores de chave de `principal`, consulte Valores de chave de principal.

- Availability (Disponibilidade): essa chave é incluída no contexto da solicitação para todas as solicitações, incluindo as anônimas.
- Tipo de dados: String
- Tipo de valor: valor único

aws:userid

Use essa chave para comparar o identificador do principal do solicitante com o ID especificado na política. Para usuários do IAM, o valor de contexto da solicitação é o ID de usuário. Para funções do IAM, esse formato de valor pode variar. Para obter detalhes sobre como as informações são exibidas para diferentes principais, consulte Como especificar uma entidade principal. Para exemplos específicos de valores de chave de `principal`, consulte Valores de chave de principal.

- Availability (Disponibilidade): essa chave é incluída no contexto da solicitação para todas as solicitações, incluindo as anônimas.
- Tipo de dados: String
- Tipo de valor: valor único

aws:username

Use essa chave para comparar o nome de usuário do solicitante com o nome de usuário especificado na política. Para obter detalhes sobre como as informações são exibidas para diferentes principais, consulte Como especificar uma entidade principal. Para exemplos específicos de valores de chave de `principal`, consulte Valores de chave de principal.

- Disponibilidade: essa chave será sempre incluída no contexto da solicitação para os usuários do IAM. Solicitações anônimas e solicitações feitas usando o Usuário raiz da conta da AWS ou perfis do IAM não incluem essa chave. As solicitações feitas usando credenciais do IAM Identity Center não incluem essa chave no contexto.
- Tipo de dados: String
- Tipo de valor: valor único

Propriedades de uma sessão de perfil

Use as chaves de condição a seguir para comparar as propriedades da sessão do perfil no momento em que a sessão foi gerada. Essas chaves de condição só estão disponíveis quando uma solicitação é feita por uma entidade principal com credenciais de sessão de perfil ou de entidade principal de usuário federado. Os valores dessas chaves de condição estão incorporados no token de sessão do perfil.

Um perfil é um tipo de entidade principal. Você também pode usar as chaves de condição da seção Propriedades da entidade principal para avaliar as propriedades de um perfil quando um perfil está fazendo uma solicitação.

aws:AssumedRoot

Use essa chave para verificar se a solicitação foi feita usando AssumeRoot. `AssumeRoot` retorna credenciais de curto prazo para uma sessão de usuário-raiz privilegiado que pode ser usada para realizar ações privilegiadas nas contas-membro em sua organização. Para obter mais informações, consulte Gerencie centralmente o acesso raiz para contas-membro.

- Disponibilidade: esta chave é incluída no contexto da solicitação somente quando a entidade principal usa credenciais de AssumeRoot para fazer a solicitação.
- Tipo de dados: Booleano
- Tipo de valor: valor único

No exemplo a seguir, quando usado como uma política de controle de serviço, impede o uso das credenciais de longo prazo de um usuário-raiz em uma conta de membro do AWS Organizations. A política não impede que as sessões de `AssumeRoot` realizem as ações permitidas por uma sessão de `AssumeRoot`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement":[
       {
          "Effect":"Deny",
          "Action":"*",
          "Resource": "*",
```

```
         "Condition":{
            "ArnLike":{
               "aws:PrincipalArn":[
                  "arn:aws:iam::*:root"
               ]
            },
            "Null":{
               "aws:AssumedRoot":"true"
            }
         }
      }
   ]
}
```

aws:FederatedProvider

Use essa chave para comparar o provedor de identidade (IdP) emissor da entidade principal com o IdP especificado na política. Isso significa que um perfil do IAM foi assumido usando a operação `AssumeRoleWithWebIdentity` do AWS STS. Quando as credenciais temporárias da sessão de função resultante são usadas para fazer uma solicitação, o contexto da solicitação identifica o IdP que autenticou a identidade federada original.

- Disponibilidade: essa chave está presente na sessão de perfil de um perfil que foi assumido usando o provedor OpenID Connect (OIDC) e na política de confiança do perfil quando um provedor OIDC está é usado para chamar `AssumeRoleWithWebIdentity`.
- Tipo de dados: string*
- Tipo de valor: valor único

* O tipo de dados depende do seu IdP:

- Se você estiver usando um IdP integrado da AWS, como o Amazon Cognito, o valor da chave será uma string. O valor da chave pode ser semelhante a: `cognito-identity.amazonaws.com`.
- Se você estiver usando um IdP que não esteja incorporado à AWS, como o GitHub ou o Amazon EKS, o valor da chave será um ARN. O valor da chave pode ser semelhante a: `arn:aws:iam::`*`111122223333`*`:oidc-provider/oidc.eks.`*`region`*`.amazonaws.com/id/`*`OIDC_Provider_ID`*.

Para obter mais informações sobre IdPs externos e `AssumeRoleWithWebIdentity`, consulte Cenários comuns. Para obter mais informações, consulte Entidades principais da sessão de função.

aws:TokenIssueTime

Use essa chave para comparar a data e a hora em que as credenciais de segurança temporárias foram emitidas com a data e a hora especificadas na política.

- Disponibilidade: esta chave é incluída no contexto da solicitação somente quando a entidade principal usa credenciais temporárias para fazer a solicitação. A chave não está presente em solicitações da AWS CLI, da API da AWS ou do AWS SDK que são feitas usando chaves de acesso.
- Tipo de dados: Data
- Tipo de valor: valor único

Para saber quais serviços oferecem suporte a credenciais de segurança temporárias, consulte AWSServiços da que funcionam com o IAM.

aws:SignInSessionArn

Use esta chave para comparar sua sessão de login OAuth da AWS com a sessão especificada na política.

Quando você usa um fluxo baseado em OAuth, como login AWS CLI (`aws login`) ou servidor MCP da AWS, o login da AWS inclui um ARN de sessão de login nas credenciais emitidas. AWS O login propaga esse ARN para solicitações subsequentes feitas usando tokens de acesso OAuth. Com essa chave, você pode correlacionar a atividade da API com a sessão de login de origem e aplicar políticas do IAM a sessões individuais.

Para fluxos OAuth interativos, o token de atualização determina o ciclo de vida do ARN da sessão de login. Um novo token de acesso emitido durante uma atualização continua carregando o mesmo ARN da sessão de login.

Para fluxos OAuth não interativos, o token de acesso determina o ciclo de vida do ARN da sessão de login. Cada novo token de acesso contém um novo ARN da sessão de login.

- Disponibilidade: esta chave é incluída no contexto da solicitação quando a solicitação se origina de uma sessão OAuth de login da AWS. Essa chave não está disponível para sessões de console.
- Tipo de dados: ARN

- Tipo de valor: valor único

Utilize operadores ARN em vez de operadores string ao comparar ARNs.

A política a seguir nega todas as ações associadas a uma sessão de login específica da AWS, embora as solicitações de outras sessões ativas continuem:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenySpecificSignInSession",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:SignInSessionArn": "arn:aws:signin:us-east-1:111122223333:session/
session-id"
        }
      }
    }
  ]
}
```

aws:MultiFactorAuthAge

Use essa chave para comparar o número de segundos desde que o principal solicitante foi autorizado usando MFA com o número especificado na política. Para obter mais informações sobre MFA, consulte AWSCódigo da autenticação multifator no IAM da.

> **Important**
>
> Essa chave de condição não está presente em identidades federadas ou solicitações feitas usando chaves de acesso para assinar solicitações da AWS CLI, da API da AWS ou do AWS SDK. Para saber mais sobre como adicionar proteção com MFA às operações de API com credenciais de segurança temporárias, consulte Acesso seguro à API com a MFA.
> Para verificar se a MFA é usada para validar identidades federadas do IAM, você pode transmitir o método de autenticação do seu provedor de identidade para a AWS como uma tag de sessão. Para obter detalhes, consulte Passar tags de sessão no AWS STS. Para aplicar a MFA às identidades do Centro de Identidade do IAM, você pode habilitar atributos

de controle de acesso para transmitir uma declaração SAML com o método de autenticação do seu provedor de identidade ao Centro de Identidade do IAM.

- Disponibilidade: esta chave é incluída no contexto da solicitação somente quando a entidade principal usa credenciais de segurança temporárias para fazer a solicitação. Políticas com condições de MFA podem ser anexadas a:
  - Um usuário ou grupo do IAM
  - Um recurso como um bucket do Amazon S3, uma fila do Amazon SQS ou um tópico do Amazon SNS
  - A política de confiança de uma função do IAM que pode ser assumida por um usuário
- Tipos de dados: Numérico
- Tipo de valor: valor único

aws:MultiFactorAuthPresent

Use essa chave para verificar se a autenticação multifator (MFA) foi usada para validar as credenciais de segurança temporárias que fizeram a solicitação.

Important

Essa chave de condição não está presente em identidades federadas ou solicitações feitas usando chaves de acesso para assinar solicitações da AWS CLI, da API da AWS ou do AWS SDK. Para saber mais sobre como adicionar proteção com MFA às operações de API com credenciais de segurança temporárias, consulte Acesso seguro à API com a MFA.
Para verificar se a MFA é usada para validar identidades federadas do IAM, você pode transmitir o método de autenticação do seu provedor de identidade para a AWS como uma tag de sessão. Para obter detalhes, consulte Passar tags de sessão no AWS STS. Para aplicar a MFA às identidades do Centro de Identidade do IAM, você pode habilitar atributos de controle de acesso para transmitir uma declaração SAML com o método de autenticação do seu provedor de identidade ao Centro de Identidade do IAM.

- Disponibilidade: esta chave é incluída no contexto da solicitação somente quando a entidade principal usa credenciais temporárias para fazer a solicitação. Políticas com condições de MFA podem ser anexadas a:

  - Um usuário ou grupo do IAM
  - Um recurso como um bucket do Amazon S3, uma fila do Amazon SQS ou um tópico do Amazon SNS
  - A política de confiança de uma função do IAM que pode ser assumida por um usuário
- Tipo de dados: Booleano
- Tipo de valor: valor único

As credenciais temporárias são usadas para autenticar perfis do IAM e usuários do IAM com tokens temporários de AssumeRole ou GetSessionToken e usuários do Console de gerenciamento da AWS.

As chaves de acesso do usuário do IAM são credenciais de longo prazo, mas, em alguns casos, a AWS cria credenciais temporárias em nome dos usuários do IAM para realizar operações. Nesses casos, a chave `aws:MultiFactorAuthPresent` está presente na solicitação e definida como um valor de `false`. Há dois casos comuns em que isso pode acontecer:

- Os usuários do IAM no Console de gerenciamento da AWS inconscientemente usam credenciais temporárias. Os usuários fazem login no console do usando seu nome de usuário e senha, que são credenciais de longo prazo. No entanto, em segundo plano, o console gera credenciais temporárias em nome do usuário.
- Se um usuário do IAM faz uma chamada para um produto da AWS, o produto reutiliza as credenciais do usuário para fazer outra solicitação a outro serviço. Por exemplo, ao chamar o Athena para acessar um bucket do Amazon S3 ou ao usar o CloudFormation para criar uma instância do Amazon EC2. Para a solicitação subsequente, a AWS usa credenciais temporárias.

Para saber quais serviços oferecem suporte a credenciais de segurança temporárias, consulte AWSServiços da que funcionam com o IAM.

A chave `aws:MultiFactorAuthPresent` nunca está presente quando uma API ou um comando da CLI é chamado com credenciais de longo prazo, como pares de chave de acesso. Portanto, recomendamos que, ao verificar essa chave, você use as versões `...IfExists` dos operadores de condição.

É importante compreender que o seguinte elemento `Condition` não é uma maneira confiável de verificar se uma solicitação é autenticada usando MFA.

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
```

```
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Essa combinação do efeito `Deny`, do elemento `Bool` e do valor `false` nega as solicitações que podem ser autenticadas usando MFA, mas que não foram. Isso se aplica apenas a credenciais temporárias que oferecem suporte usando MFA. Essa declaração não nega o acesso a solicitações que são feitas usando credenciais de longo prazo ou a solicitações que são autenticadas usando MFA. Use este exemplo com cuidado, pois a lógica é complicada e não testa se a autenticação por MFA foi realmente utilizada.

Além disso, não use a combinação do efeito `Deny`, do elemento `Null` e `true` porque ela se comporta da mesma forma e a lógica é ainda mais complicada.

Combinação recomendada

Recomendamos usar o operador `BoolIfExists` para verificar se um solicitação foi autenticada usando MFA.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Essa combinação de `Deny`, `BoolIfExists` e `false` nega solicitações que não são autenticados usando MFA. Especificamente, ela nega solicitações de credenciais temporárias que não incluem MFA. Ele também nega solicitações que são feitas usando credenciais de longo prazo, como operações da AWS CLI ou da API da AWS feitas usando chaves de acesso. O operador `*IfExists` verificará a presença da chave `aws:MultiFactorAuthPresent` e se ela poderia ou não estar presente, conforme indicado pela sua existência. Use essa opção quando quiser recusar qualquer solicitação que não tenha sido autenticada usando MFA. Isso é mais seguro, mas pode danificar um código ou script que use chaves de acesso para acessar a AWS CLI ou a API da AWS.

Combinações alternativas

Também é possível usar o operador `BoolIfExists` para permitir solicitações autenticadas por MFA e solicitações da AWS CLI ou da API da AWS que são feitas usando credenciais de longo prazo.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Essa condição corresponde à condição se a chave existe e está presente ou se a chave não existe. Essa combinação de `Allow`, `BoolIfExists`, e `true` permite as solicitações autenticadas

usando MFA ou as solicitações que não podem ser autenticadas usando MFA. Isso significa que as operações da AWS CLI, da API da AWS e do AWS SDK são permitidas quando o solicitante usa as chaves de acesso de longo prazo dele. Essa combinação não permite solicitações de credenciais temporárias que poderiam, mas não incluem MFA.

Quando você cria uma política usando o editor visual do console do IAM e seleciona MFA required (MFA obrigatória), essa combinação é aplicada. Essa configuração requer MFA para acesso ao console, mas permite acesso programático sem MFA.

Como alternativa, você pode usar o operador `Bool` para permitir solicitações programáticas e de console somente quando a autenticação for feita usando MFA.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Essa combinação de `Allow`, `Bool` e `true` permite apenas solicitações autenticadas por MFA. Isso se aplica apenas a credenciais temporárias que oferecem suporte usando MFA. Essa declaração não permite o acesso a solicitações que são feitas usando chaves de acesso de longo prazo ou a solicitações feitas usando credenciais temporárias sem MFA.

Não use um elemento de política seguinte ao seguinte para verificar se a chave de MFA está presente:

```
#####   WARNING: USE WITH CAUTION   #####

"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Essa combinação do efeito `Allow`, do elemento `Null` e do valor `false` permite apenas solicitações que podem ser autenticadas usando MFA, independentemente de a solicitação ser realmente autenticada. Isso permite todas as solicitações que são feitas usando credenciais temporárias e nega o acesso para credenciais de longo prazo. Use este exemplo com cuidado porque ele não testa se a autenticação por MFA foi realmente utilizada.

aws:ChatbotSourceArn

Use essa chave para comparar o ARN da configuração do chat de origem definido pela entidade principal com o ARN da configuração do chat que você especificou na política do perfil do IAM associada à configuração do canal. É possível autorizar solicitações com base na sessão de perfil assumida iniciada pelo Amazon Q Developer em aplicações de chat.

- Disponibilidade: essa chave é incluída no contexto da solicitação pelo serviço Amazon Q Developer em aplicações de chat sempre que uma sessão de perfil é assumida. O valor da chave é o ARN da configuração do chat, como quando você executa um comando da AWS CLI em um canal de chat.
- Tipo de dados: ARN
- Tipo de valor: valor único
- Exemplo de valor – `arn:aws::chatbot::123456789021:chat-configuration/slack-channel/private_channel`

A política a seguir nega as solicitações de put do Amazon S3 no bucket especificado para todas as solicitações originadas em canal do Slack.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleS3Deny",
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnLike": {
                      "aws:ChatbotSourceArn": "arn:aws:chatbot::*:chat-
configuration/slack-channel/*"
                }
            }
        }
    ]
}
```

aws:Ec2InstanceSourceVpc

Essa chave identifica a VPC para a qual as credenciais de perfil do IAM do Amazon EC2 foram entregues. É possível usar essa chave em uma política com a chave global `aws:SourceVPC` para verificar se uma chamada é feita por uma VPC (`aws:SourceVPC`) que corresponde à VPC para a qual a credencial foi entregue (`aws:Ec2InstanceSourceVpc`).

- Disponibilidade: esta chave será incluída no contexto da solicitação sempre que o solicitante assinar solicitações com uma credencial de perfil do Amazon EC2. Pode ser usada em políticas do IAM, políticas de controle de serviços, políticas de endpoint da VPC e políticas de recursos.
- Tipo de dados: String
- Tipo de valor: valor único

Essa chave pode ser usada com valores de identificador de VPC, porém é mais útil quando usada como uma variável combinada com a chave de contexto `aws:SourceVpc`. A chave de contexto `aws:SourceVpc` será incluída no contexto da solicitação somente se o solicitante usar um endpoint da VPC para fazer a solicitação. Usar `aws:Ec2InstanceSourceVpc` com `aws:SourceVpc` permite que você utilize `aws:Ec2InstanceSourceVpc` de forma mais ampla, pois compara valores que normalmente são alterados juntos.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

No exemplo acima, o acesso será negado se o valor de `aws:SourceVpc` não for igual ao valor de `aws:Ec2InstanceSourceVpc`. A instrução de política é limitada somente aos perfis usados como perfis de instância do Amazon EC2, testando a existência da chave de condição `ec2:SourceInstanceARN`.

A política usa `aws:ViaAWSService` para permitir que a AWS autorize solicitações quando as solicitações são feitas em nome de seus perfis de instância do Amazon EC2. Por exemplo, quando você faz uma solicitação de uma instância do Amazon EC2 para um bucket criptografado do Amazon S3, o Amazon S3 faz uma chamada para o AWS KMS por você. Algumas das chaves não estão presentes quando a solicitação é feita ao AWS KMS.

aws:Ec2InstanceSourcePrivateIPv4

Esta chave identifica o endereço IPv4 privado da interface de rede elástica primária ao qual as credenciais de perfil do IAM do Amazon EC2 foram entregues. É necessário usar essa chave de condição com sua chave complementar `aws:Ec2InstanceSourceVpc` para certificar-se de que você tenha uma combinação global exclusiva de ID da VPC e IP privado de origem. Use essa chave com `aws:Ec2InstanceSourceVpc` para certificar-se de que uma solicitação tenha sido feita com base no mesmo endereço IP privado para o qual as credenciais foram entregues.

- Disponibilidade: esta chave será incluída no contexto da solicitação sempre que o solicitante assinar solicitações com uma credencial de perfil do Amazon EC2. Pode ser usada em políticas do IAM, políticas de controle de serviços, políticas de endpoint da VPC e políticas de recursos.
- Tipo de dados: Endereço IP
- Tipo de valor: valor único

> **Important**
>
> Essa chave não deve ser usada sozinha em uma instrução `Allow`. Por definição, os endereços IP privados não são globalmente exclusivos. É necessário usar a chave `aws:Ec2InstanceSourceVpc` toda vez que usar a chave `aws:Ec2InstanceSourcePrivateIPv4` para especificar a VPC de onde suas credenciais de instância do Amazon EC2 podem ser usadas.

O exemplo a seguir é uma política de controle de serviços (SCP) que nega acesso a todos os recursos, a menos que a solicitação chegue por meio de um endpoint da VPC na mesma VPC das

credenciais do perfil. Neste exemplo, `aws:Ec2InstanceSourcePrivateIPv4` limita a origem da credencial a uma instância específica com base no IP de origem.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

aws:SourceIdentity

Use essa chave para comparar a identidade-fonte definida pela entidade principal com a identidade-fonte especificada na política.

- Disponibilidade: essa chave é incluída no contexto da solicitação depois que uma identidade-fonte é definida quando uma função é assumida usando qualquer comando de assumir função da CLI do AWS STS ou a operação `AssumeRole` da API do AWS STS.
- Tipo de dados: String
- Tipo de valor: valor único

É possível usar essa chave em uma política para permitir ações na AWS executadas por entidades de segurança que definiram uma identidade-fonte ao assumir um perfil. A atividade para a identidade-fonte especificada da função aparece no AWS CloudTrail. Isso torna mais fácil para os administradores determinar quem ou o que executou ações com uma função na AWS.

Ao contrário de `sts:RoleSessionName`, após a definição da identidade-fonte, o valor não pode ser alterado. Ele estará presente no contexto da solicitação para todas as ações executadas pela função. O valor persiste nas sessões de função subsequentes quando você usa as credenciais da sessão para assumir outra função. Assumir uma função de outra é chamado de encadeamento de funções.

A chave `sts:SourceIdentity` está presente na solicitação quando a entidade principal inicialmente define uma identidade-fonte enquanto assume um perfil usando qualquer comando de assumir perfil da CLI do AWS STS ou operação da API `AssumeRole` do AWS STS. A chave `aws:SourceIdentity` está presente na solicitação para todas as ações executadas com uma sessão de função que possui um conjunto de identidade-fonte.

A política de confiança de perfil a seguir para `CriticalRole` na conta `111122223333` contém uma condição para `aws:SourceIdentity` que impede que uma entidade principal sem uma identidade-fonte definida como Saanvi ou Diego assuma o perfil.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
```

```
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Para saber mais sobre como usar informações de identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

ec2:RoleDelivery

Use essa chave para comparar a versão do serviço de metadados da instância na solicitação assinada com as credenciais do perfil do IAM para o Amazon EC2. O serviço de metadados da instância faz distinção entre as solicitações do IMDSv1 e do IMDSv2 com base na presença dos cabeçalhos `PUT` ou `GET` , que são exclusivos do IMDSv2, nessa solicitação.

- Disponibilidade: esta chave será incluída no contexto da solicitação sempre que a sessão de perfil for criada por uma instância do Amazon EC2.
- Tipos de dados: Numérico
- Tipo de valor: valor único
- Valores de exemplo – 1.0, 2.0

É possível configurar o Serviço de metadados de instância (IMDS) em cada instância de modo que o código ou os usuários locais devam usar o IMDSv2. Quando você especifica que o IMDSv2 deve ser usado, o IMDSv1 não funciona mais.

- Serviço de metadados da instância versão 1 (IMDSv1): um método de solicitação/resposta
- Serviço de metadados da instância versão 2 (IMDSv2): um método orientado a sessões

Para obter informações sobre como configurar sua instância para usar o IMDSv2, consulte Configurar as opções de metadados da instância.

No exemplo a seguir, o acesso será negado se o valor ec2:RoleDelivery no contexto da solicitação for 1.0 (IMDSv1). Essa instrução/política pode ser aplicada de modo geral porque, se a solicitação não for assinada por credenciais de perfil do Amazon EC2, ela não terá efeito.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Para obter mais informações, consulte Exemplo de políticas para trabalhar com metadados de instâncias.

ec2:SourceInstanceArn

Use essa chave para comparar o ARN da instância da qual a sessão do perfil foi gerada.

- Disponibilidade: esta chave será incluída no contexto da solicitação sempre que a sessão de perfil for criada por uma instância do Amazon EC2.
- Tipo de dados: ARN
- Tipo de valor: valor único
- Valor de exemplo: arn:aws:ec2:us-west-2:111111111111:instance/instance-id

Para obter exemplos de políticas, consulte Permitir que uma instância específica visualize recursos em outros serviços da AWS.

glue:RoleAssumedBy

O serviço AWS Glue define essa chave de condição para cada solicitação à API da AWS em que AWS Glue faz uma solicitação usando um perfil de serviço em nome do cliente (não por meio de um endpoint de trabalho ou desenvolvedor, mas diretamente pelo serviço AWS Glue). Use essa chave para verificar se uma chamada para um recurso da AWS é proveniente do serviço AWS Glue.

- Disponibilidade: essa chave será incluída no contexto da solicitação quando o AWS Glue fizer uma solicitação usando um perfil de serviço em nome do cliente.
- Tipo de dados: String
- Tipo de valor: valor único
- Valor de exemplo: essa chave é sempre definida como `glue.amazonaws.com`.

O exemplo a seguir adiciona uma condição para permitir que o serviço AWS Glue obtenha um objeto de um bucket do Amazon S3.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

glue:CredentialIssuingService

O serviço AWS Glue define essa chave para cada solicitação à API da AWS usando um perfil de serviço proveniente de um endpoint de trabalho ou desenvolvedor. Use essa chave para verificar se uma chamada para um recurso da AWS é proveniente de um endpoint de trabalho ou desenvolvedor do AWS Glue.

- Disponibilidade: essa chave será incluída no contexto da solicitação quando o AWS Glue fizer uma solicitação proveniente de um endpoint de trabalho ou desenvolvedor.

- Tipo de dados: String
- Tipo de valor: valor único
- Valor de exemplo: essa chave é sempre definida como `glue.amazonaws.com`.

O exemplo seguir adiciona uma condição vinculada a um perfil do IAM usado por um trabalho do AWS Glue. Isso garante que determinadas ações sejam permitidas/negadas dependendo de a sessão de perfil ser ou não usada para um ambiente de runtime de trabalho do AWS Glue.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

codebuild:BuildArn

Essa chave identifica o ARN da compilação do CodeBuild à qual as credenciais do perfil do IAM foram fornecidas. Use essa chave para verificar se uma chamada para um recurso da AWS é proveniente de uma determinada compilação do CodeBuild.

> **Note**
>
> O valor total de `codebuild:BuildArn` não é conhecido com antecedência porque contém o ID da compilação gerado dinamicamente.

- Disponibilidade: essa chave é incluída no contexto da solicitação sempre que uma solicitação é feita por um perfil assumido pelo CodeBuild.
- Tipo de dados: ARN
- Tipo de valor: valor único
- Exemplo de valor: arn:aws:codebuild:us-east-1:123456789012:build/MyBuildProject:12345678-1234-1234-1234-123456789012

O exemplo a seguir permite que uma determinada compilação do CodeBuild tenha acesso de `s3:GetObject` ao bucket especificado.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnLike": {
            "codebuild:BuildArn": "arn:aws:codebuild:us-east-1:123456789012:build/
MyBuildProject:*"
        }
    }
}
```

codebuild:ProjectArn

Essa chave identifica o ARN do projeto do CodeBuild ao qual as credenciais do perfil do IAM foram fornecidas. Use essa chave para verificar se uma chamada para um recurso da AWS é proveniente de um determinado projeto do CodeBuild.

- Disponibilidade: essa chave é incluída no contexto da solicitação sempre que uma solicitação é feita por um perfil assumido pelo CodeBuild.
- Tipo de dados: ARN
- Tipo de valor: valor único
- Exemplo de valor: arn:aws:codebuild:us-east-1:123456789012:project/MyBuildProject

O exemplo a seguir permite que qualquer compilação de um determinado projeto do CodeBuild tenha acesso de `s3:GetObject` ao bucket especificado.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "ArnEquals": {
            "codebuild:ProjectArn": "arn:aws:codebuild:us-east-1:123456789012:project/
MyBuildProject"
        }
```

```
    }
}
```

lambda:SourceFunctionArn

Use essa chave para identificar o ARN da função do Lambda para o qual as credenciais do perfil do IAM foram entregues. O serviço Lambda define essa chave para cada solicitação à API da AWS proveniente do ambiente de execução da sua função. Use essa chave para verificar se uma chamada para um recurso da AWS é proveniente do código de uma função específica do Lambda. O Lambda também define essa chave para algumas solicitações feitas fora do ambiente de execução, como gravar logs no CloudWatch e enviar rastreamentos para o X-Ray.

- Disponibilidade: essa chave será incluída no contexto da solicitação sempre que o código da função do Lambda for invocado.
- Tipo de dados: ARN
- Tipo de valor: valor único
- Valor de exemplo: arn:aws:lambda:us-east-1:123456789012:function:TestFunction

O exemplo a seguir permite que uma função do Lambda específica tenha acesso de `s3:PutObject` ao bucket especificado.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
```

```
}
```

Para obter mais informações, consulte Trabalhar com credenciais do ambiente de execução do Lambda no Guia do desenvolvedor do AWS Lambda.

ssm:SourceInstanceArn

Use essa chave para identificar o ARN da instância gerenciada do AWS Systems Manager para o qual as credenciais do perfil do IAM foram entregues. Essa chave está não está presente quando a solicitação é proveniente instância gerenciada com um perfil do IAM associado a um perfil de instância do Amazon EC2.

- Disponibilidade: essa chave será incluída no contexto da solicitação sempre que as credenciais do perfil forem entregues a uma instância gerenciada do AWS Systems Manager.
- Tipo de dados: ARN
- Tipo de valor: valor único
- Valor de exemplo: arn:aws:ec2:us-west-2:111111111111:instance/instance-id

identitystore:UserId

Use essa chave para comparar a identidade da força de trabalho do IAM Identity Center na solicitação assinada com a identidade especificada na política.

- Disponibilidade: essa chave é incluída quando o chamador da solicitação é um usuário no IAM Identity Center.
- Tipo de dados: String
- Tipo de valor: valor único
- Valor de exemplo: 94482488-3041-7026-18f3-be45837cd0e4

É possível encontrar o UserId de um usuário no IAM Identity Center fazendo uma solicitação à API GetUserId usando a AWS CLI, a API da AWS ou AWS SDK.

Propriedades da rede

Use as chaves de condição a seguir para comparar detalhes sobre a rede da qual a solicitação é proveniente ou foi passada por meio das propriedades de rede que você especificar na política.

aws:SourceIp

Use essa chave para comparar o endereço IP do solicitante com o endereço IP especificado na política. A chave de condição `aws:SourceIp` só pode ser usada para intervalos de endereços IP públicos.

- Disponibilidade: essa chave será incluída no contexto da solicitação, exceto quando o solicitante usar um endpoint da VPC para fazer a solicitação.
- Tipo de dados: Endereço IP
- Tipo de valor: valor único

A chave de condição `aws:SourceIp` pode ser usada em uma política para permitir que os principais façam solicitações somente em um intervalo de IP especificado.

> **Note**
>
> O `aws:SourceIp` oferece suporte a endereços IP para IPv4 e IPv6 ou a um intervalo de endereços IP. Para obter uma lista de Serviços da AWS com suporte para IPv6, consulte Serviços da AWS que oferecem suporte a IPv6 no Guia do usuário da Amazon VPC.

Por exemplo, é possível anexar a política baseada em identidade apresentada a seguir a um perfil do IAM. Essa política permite que o usuário coloque objetos no bucket `amzn-s3-demo-bucket3` do Amazon S3 caso faça a chamada no intervalo de endereços IPv4 especificado. Essa política também permite um serviço da AWS que use Sessões de acesso direto faça essa operação em seu nome.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
```

```
                }
            }
        }
    ]
}
```

Caso seja necessário restringir o acesso de redes que oferecem suporte ao endereçamento IPv4 e IPv6, você poderá incluir os endereços IPv4 e IPv6 ou os intervalos de endereços IP na condição da política do IAM. A política baseada em identidade a seguir permitirá que o usuário coloque objetos no bucket `amzn-s3-demo-bucket3` do Amazon S3 caso ele faça a chamada em intervalos de endereços IPv4 ou IPv6 especificados. Antes de incluir intervalos de endereços IPv6 em sua política do IAM, verifique se o AWS service (Serviço da AWS) com o qual você está trabalhando oferece suporte a IPv6. Para obter uma lista de Serviços da AWS com suporte para IPv6, consulte Serviços da AWS que oferecem suporte a IPv6 no Guia do usuário da Amazon VPC.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Se a solicitação vier de um host que use um endpoint do Amazon VPC, a chave `aws:SourceIp` não estará disponível. Em vez disso, use uma chave específica da VPC, como aws:VpcSourceIp. Para

obter mais informações sobre o uso de endpoints da VPC, consulte Gerenciamento de identidades e acesso para endpoints da VPC e serviços de endpoint da VPC no Guia do AWS PrivateLink.

> **Note**
>
> Quando Serviços da AWS fazem chamadas para outros Serviços da AWS em seu nome (chamadas entre serviços), um determinado contexto de autorização específico da rede é removido. Se sua política usa essa chave de condição com declarações `Deny`, as entidades principais do AWS service (Serviço da AWS) podem ser bloqueadas involuntariamente. Para permitir que os Serviços da AWS funcionem adequadamente enquanto mantém seus requisitos de segurança, exclua as entidades principais de serviço de suas declarações `Deny` adicionando a chave de condição `aws:PrincipalIsAWSService` com um valor de `false`.

aws:SourceVpc

Use essa chave para verificar se a solicitação trafega pela VPC à qual o endpoint da VPC está conectado. Em uma política, é possível usar essa chave para permitir acesso apenas a uma VPC específica. Para mais informações, consulte Restringir acesso a uma VPC específica no Guia do usuário do Amazon Simple Storage Service.

- Disponibilidade: essa chave é incluída no contexto da solicitação somente se o solicitante usar um endpoint da VPC para fazer a solicitação.
- Tipo de dados: String
- Tipo de valor: valor único

Em uma política, você pode usar essa chave para permitir ou restringir o acesso a uma VPC específica.

Por exemplo, você pode anexar a seguinte política baseada em identidade a um perfil do IAM para negar `PutObject` ao bucket `amzn-s3-demo-bucket3` do Amazon S3, a menos que a solicitação seja feita por um ID da VPC especificada ou por Serviços da AWS que usem sessões de acesso direto (FAS) para fazer solicitações em nome do perfil. Ao contrário de aws:SourceIp, você deve usar aws:ViaAWSService ou aws:CalledVia para permitir solicitações de FAS, porque a VPC de origem da solicitação inicial não é preservada.

> **Note**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "PutObjectIfNotVPCID",
      "Effect": "Deny",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket3/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-1234567890abcdef0"
        },
        "Bool": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

> **Note**
>
> A AWS recomenda usar `aws:SourceVpcArn` em vez de `aws:SourceVpc` se `aws:SourceVpcArn` for suportado pelo serviço de destino. Consulte aws:SourceVpcArn para ver a lista de serviços compatíveis.

> **Note**
>
> Quando Serviços da AWS fazem chamadas para outros Serviços da AWS em seu nome (chamadas entre serviços), um determinado contexto de autorização específico da rede é

removido. Se sua política usa essa chave de condição com declarações `Deny`, as entidades principais do AWS service (Serviço da AWS) podem ser bloqueadas involuntariamente. Para permitir que os Serviços da AWS funcionem adequadamente enquanto mantém seus requisitos de segurança, exclua as entidades principais de serviço de suas declarações `Deny` adicionando a chave de condição `aws:PrincipalIsAWSService` com um valor de `false`.

aws:SourceVpcArn

Use essa chave para verificar o ARN da VPC por meio da qual uma solicitação foi feita usando um endpoint da VPC. Essa chave retorna o ARN da VPC à qual o endpoint da VPC está anexado.

- Disponibilidade: esta chave está incluída no contexto da solicitação para serviços compatíveis quando uma solicitação é feita por meio de um endpoint da VPC. A chave não está incluída para solicitações feitas por meio de endpoints de serviço público. Os serviços a seguir oferecem suporte a essa chave:
  - AWS App Runner (prefixo: `apprunner`)
  - AWS Application Discovery Service (prefixo: `discovery`)
  - Amazon Athena (prefixo: `athena`)
  - AWS Cloud Map (prefixo: `servicediscovery`)
  - Amazon CloudWatch Application Insights (prefixo: `applicationinsights`)
  - AWS CloudFormation (prefixo: `cloudformation`)
  - Amazon Comprehend Medical (prefixo: `comprehendmedical`)
  - AWS Compute Optimizer (prefixo: `compute-optimizer`)
  - Amazon; Elastic Container Registry (prefixo: `ecr`)
  - Amazon Elastic Container Service (prefixo: `ecs`)
  - Amazon Kinesis Analytics (prefixo: `kinesisanalytics`)
  - Amazon Route 53 (prefixo: `route53`)
  - AWS DataSync (prefixo: `datasync`)
  - Amazon Elastic Block Store (prefixo: `ebs`)
  - Agendador do Amazon EventBridge (prefixo: `scheduler`)
  - Amazon Data Firehose (prefixo: `firehose`)
  - AWS HealthImaging (prefixo: `medical-imaging`)
  - AWS HealthLake (prefixo: `healthlake`)

- AWS HealthOmics (prefixo: `omics`)
- AWS Identity and Access Management(exceto para a ação `iam:PassRole`) (prefixo: `iam`)
- AWS IoT FleetWise (prefixo: `iotfleetwise`)
- AWS IoT Wireless (prefixo: `iotwireless`)
- AWS Key Management Service (prefixo: `kms`)
- AWS Lambda (prefixo: `lambda`)
- AWS Payment Cryptography (prefixo: `payment-cryptography`)
- Amazon Polly (prefixo: `polly`)
- Autoridade de certificação privada da AWS (prefixo: `acm-pca`)
- AWS Lixeira da (prefixo: `rbin`)
- Amazon Rekognition (prefixo: `rekognition`)
- Service Quotas (prefixo: `servicequotas`)
- Amazon Simple Storage Service (prefixo: `s3`)
- AWS Storage Gateway (prefixo: `storagegateway`)
- AWS Systems Manager Incident Manager Contatos da (prefixo: `ssm-contacts`)
- Amazon Textract (prefixo: `textract`)
- Amazon Transcribe (prefixo: `transcribe`)
- AWS Transfer Family (prefixo: `transfer`)

- Tipo de dados: ARN

  A AWS recomenda utilizar operadores ARN em vez de operadores string ao comparar ARNs.
- Tipo de valor: valor único
- Exemplo de valor – `arn:aws:ec2:us-east-1:123456789012:vpc/vpc-0e9801d129EXAMPLE`

Confira um exemplo de uma política de bucket que nega acesso a `amzn-s3-demo-bucket` e seus objetos para qualquer pessoa fora da VPC `vpc-1a2b3c4d`.

```
{
   "Version":"2012-10-17",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
```

```
      "Principal": "*",
      "Action": "s3:*",
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket",
                   "arn:aws:s3:::amzn-s3-demo-bucket/*"],
      "Condition": {
        "ArnNotEquals": {
          "aws:SourceVpcArn": "arn:aws:ec2:us-east-1:*:vpc/vpc-1a2b3c4d"
        }
      }
    }
  ]
}
```

> **Note**
>
> Quando Serviços da AWS fazem chamadas para outros Serviços da AWS em seu nome (chamadas entre serviços), um determinado contexto de autorização específico da rede é removido. Se sua política usa essa chave de condição com declarações `Deny`, as entidades principais do AWS service (Serviço da AWS) podem ser bloqueadas involuntariamente. Para permitir que os Serviços da AWS funcionem adequadamente enquanto mantém seus requisitos de segurança, exclua as entidades principais de serviço de suas declarações `Deny` adicionando a chave de condição `aws:PrincipalIsAWSService` com um valor de `false`.

aws:SourceVpce

Use essa chave para comparar o identificador do VPC endpoint da solicitação com o ID do endpoint especificado na política.

- Disponibilidade: essa chave é incluída no contexto da solicitação somente se o solicitante usar um endpoint da VPC para fazer a solicitação.
- Tipo de dados: String
- Tipo de valor: valor único

Em uma política, é possível usar essa chave para restringir o acesso a um VPC endpoint específico. Para mais informações, consulte Restringir acesso a uma VPC específica no Guia do usuário do Amazon Simple Storage Service. Assim como ao usar aws:SourceVpc, você deve usar aws:ViaAWSService ou aws:CalledVia para permitir solicitações feitas por Serviços da AWS usando

sessões de acesso direto (FAS). Isso ocorre porque o endpoint da VPC de origem da solicitação inicial não é preservado.

> **Note**
>
> Quando Serviços da AWS fazem chamadas para outros Serviços da AWS em seu nome (chamadas entre serviços), um determinado contexto de autorização específico da rede é removido. Se sua política usa essa chave de condição com declarações `Deny`, as entidades principais do AWS service (Serviço da AWS) podem ser bloqueadas involuntariamente. Para permitir que os Serviços da AWS funcionem adequadamente enquanto mantém seus requisitos de segurança, exclua as entidades principais de serviço de suas declarações `Deny` adicionando a chave de condição `aws:PrincipalIsAWSService` com um valor de `false`.

aws:VpceAccount

Use essa chave para comparar o ID da conta daAWS que possui o endpoint da VPC por meio do qual a solicitação foi feita com o ID da conta especificado na política. Essa chave de condição ajuda você a estabelecer controles de perímetro de rede, garantindo que as solicitações cheguem por meio de endpoints da VPC pertencentes a contas específicas.

- Disponibilidade: essa chave será incluída no contexto da solicitação quando a solicitação for feita usando um endpoint da VPC. A chave não está incluída para solicitações feitas por meio de endpoints de serviço público.

  Os serviços a seguir oferecem suporte a essa chave:

  - AWS App Runner (prefixo: `apprunner`)
  - AWS Application Discovery Service (prefixo: `discovery`)
  - Amazon Athena (prefixo: `athena`)
  - AWS Cloud Map (prefixo: `servicediscovery`)
  - Amazon CloudWatch Application Insights (prefixo: `applicationinsights`)
  - AWS CloudFormation (prefixo: `cloudformation`)
  - Amazon Comprehend Medical (prefixo: `comprehendmedical`)
  - AWS Compute Optimizer (prefixo: `compute-optimizer`)
  - Amazon; Elastic Container Registry (prefixo: `ecr`)
  - Amazon Elastic Container Service (prefixo: `ecs`)

- Amazon Kinesis Analytics (prefixo: `kinesisanalytics`)
- Amazon Route 53 (prefixo: `route53`)
- AWS DataSync (prefixo: `datasync`)
- Amazon DynamoDB (prefixo: `dynamodb`)
- Amazon Elastic Block Store (prefixo: `ebs`)
- Agendador do Amazon EventBridge (prefixo: `scheduler`)
- Amazon Data Firehose (prefixo: `firehose`)
- Amazon GameLift (prefixo: `gamelift`)
- AWS HealthImaging (prefixo: `medical-imaging`)
- AWS HealthLake (prefixo: `healthlake`)
- AWS HealthOmics (prefixo: `omics`)
- AWS Identity and Access Management(exceto para a ação `iam:PassRole`) (prefixo: `iam`)
- AWS IoT FleetWise (prefixo: `iotfleetwise`)
- AWS IoT Wireless (prefixo: `iotwireless`)
- AWS Key Management Service (prefixo: `kms`)
- AWS Lambda (prefixo: `lambda`)
- AWS Payment Cryptography (prefixo: `payment-cryptography`)
- Amazon Polly (prefixo: `polly`)
- Autoridade de certificação privada da AWS (prefixo: `acm-pca`)
- AWS Lixeira da (prefixo: `rbin`)
- Amazon Rekognition (prefixo: `rekognition`)
- Service Quotas (prefixo: `servicequotas`)
- Amazon Simple Storage Service (prefixo: `s3`)
- AWS Shield (prefixo: `shield`)
- AWS Storage Gateway (prefixo: `storagegateway`)
- AWS Systems Manager Incident Manager Contatos da (prefixo: `ssm-contacts`)
- Amazon Textract (prefixo: `textract`)
- Amazon Transcribe (prefixo: `transcribe`)
- AWS Transfer Family (prefixo: `transfer`)

- Tipo de dados: string

- Tipo de valor: valor único
- Exemplo de valor – `123456789012`

Você pode usar essa chave de condição para restringir o acesso aos recursos para que as solicitações cheguem por meio de endpoints da VPC pertencentes à sua conta. O seguinte exemplo de política de bucket do Amazon S3 permite acesso quando a solicitação vem por meio de um endpoint da VPC de propriedade da conta especificada:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessToSpecificVpceAccountOnly",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/RoleName"
            },
            "Action": "s3:GetObject",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
            "Condition": {
                "StringEquals": {
                    "aws:VpceAccount": "111122223333"
                }
            }
        }
    ]
}
```

> **Note**
>
> Atualmente, essa chave de condição é suportada por um conjunto selecionado de serviços da AWS. O uso dessa chave com serviços não suportados pode levar a resultados de autorização não intencionais. Sempre especifique a chave de condição para serviços suportados em suas políticas.

Alguns serviços da AWS acessam seus recursos a partir de suas redes quando agem em seu nome. Se você usar esses serviços, precisará editar o exemplo de política acima para permitir que os serviços da AWS acessem seus recursos de fora da sua rede. Para obter mais informações sobre padrões de acesso que precisam ser considerados ao aplicar controles de acesso com base na origem da solicitação, consulte Estabeleça barreiras de proteção para permissões usando perímetros de dados.

aws:VpceOrgID

Use esta chave para comparar o identificador da organização do AWS Organizations que é proprietário do endpoint da VPC do qual a solicitação foi feita com o identificador especificado na política. Essa chave de condição fornece a abordagem mais escalável para controles de perímetro de rede, incluindo automaticamente todos os endpoints da VPC de propriedade de contas em sua organização.

- Disponibilidade: essa chave é incluída no contexto da solicitação quando uma solicitação é feita por meio de um endpoint da VPC e a conta do proprietário do endpoint da VPC é membro de uma organização da AWS. A chave não está incluída para solicitações feitas por meio de outros caminhos de rede ou quando a conta do proprietário do endpoint da VPC não faz parte de uma organização.

  Os serviços a seguir oferecem suporte a essa chave de condição:

  - AWS App Runner (prefixo: `apprunner`)
  - AWS Application Discovery Service (prefixo: `discovery`)
  - Amazon Athena (prefixo: `athena`)
  - AWS B2B Data Interchange (prefixo: `b2bi`)
  - AWS Cloud Map (prefixo: `servicediscovery`)
  - AWS CloudHSM (prefixo: `cloudhsmv2`)
  - Amazon CloudWatch Application Insights (prefixo: `applicationinsights`)
  - AWS CloudFormation (prefixo: `cloudformation`)
  - Amazon Cognito (prefixo: `cognito-identity`)
  - Amazon Comprehend Medical (prefixo: `comprehendmedical`)
  - AWS Compute Optimizer (prefixo: `compute-optimizer`)
  - AWS Database Migration Service (prefixo: `dms`)
  - AWS Dados do Directory Service (prefixo: `ds-data`)

- Amazon DynamoDB (prefixo: `dynamodb`)
- Amazon; Elastic Container Registry (prefixo: `ecr`)
- Amazon Elastic Container Service (prefixo: `ecs`)
- Amazon Kinesis Data Streams (prefixo: `kinesis`)
- Amazon Kinesis Analytics (prefixo: `kinesisanalytics`)
- Amazon Q Business (prefixo: `qbusiness`)
- Amazon Route 53 (prefixo: `route53`)
- AWS DataSync (prefixo: `datasync`)
- Amazon Elastic Block Store (prefixo: `ebs`)
- Agendador do Amazon EventBridge (prefixo: `scheduler`)
- Amazon Data Firehose (prefixo: `firehose`)
- Amazon GameLift (prefixo: `gamelift`)
- AWS HealthImaging (prefixo: `medical-imaging`)
- AWS HealthLake (prefixo: `healthlake`)
- AWS HealthOmics (prefixo: `omics`)
- AWS Identity and Access Management(exceto para a ação `iam:PassRole`) (prefixo: `iam`)
- AWS Identity Store (prefixo: `identitystore`)
- AWS IoT FleetWise (prefixo: `iotfleetwise`)
- AWS IoT TwinMaker (prefixo: `iottwinmaker`)
- AWS IoT Wireless (prefixo: `iotwireless`)
- Amazon Keyspaces (para Apache Cassandra) (prefixo: `cassandra`)
- AWS Key Management Service (prefixo: `kms`)
- AWS Lambda (prefixo: `lambda`)
- AWS Network Firewall (prefixo: `network-firewall`)
- AWS Payment Cryptography (prefixo: `payment-cryptography`)
- SMS e serviço de voz do Amazon Pinpoint (prefixo: `sms-voice`)
- Amazon Polly (prefixo: `polly`)
- AWS Price List (prefixo: `pricing`)
- Autoridade de certificação privada da AWS (prefixo: `acm-pca`)
- AWS Lixeira da (prefixo: `rbin`)

- Amazon Rekognition (prefixo: `rekognition`)
- Service Quotas (prefixo: `servicequotas`)
- Amazon Simple Email Service (SES) (prefixo: `ses`)
- AWS Shield (prefixo: `shield`)
- Amazon Simple Storage Service (prefixo: `s3`)
- Amazon Simple Queue Service (prefixo: `sqs`)
- AWS Storage Gateway (prefixo: `storagegateway`)
- AWS Systems Manager Incident Manager Contatos da (prefixo: `ssm-contacts`)
- Amazon Textract (prefixo: `textract`)
- Amazon Transcribe (prefixo: `transcribe`)
- AWS Transfer Family (prefixo: `transfer`)
- Amazon WorkMail (prefixo: `workmail`)

- Tipo de dados: string
- Tipo de valor: valor único
- Valores de exemplo – `o-a1b2c3d4e5`

O exemplo de política de controle de recursos a seguir nega acesso ao seu Amazon S3 e aos recursos da AWS Key Management Service, a menos que a solicitação venha por meio de endpoints da VPC pertencentes à organização especificada ou de redes de serviços da AWS que atuam em seu nome. Algumas organizações podem precisar editar essa política adicionalmente para atender às respectivas necessidades, por exemplo, permitir o acesso de parceiros terceirizados. Para obter mais informações sobre padrões de acesso que precisam ser considerados ao aplicar controles de acesso com base na origem da solicitação, consulte Estabeleça barreiras de proteção para permissões usando perímetros de dados.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "EnforceNetworkPerimeterVpceOrgID",
      "Effect": "Deny",
      "Principal": "*",
```

```
      "Action": [
        "s3:*",
        "kms:*"
      ],
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false",
          "aws:ViaAWSService": "false"
        },
        "StringNotEqualsIfExists": {
            "aws:VpceOrgID": "o-abcdef0123",
            "aws:PrincipalTag/network-perimeter-exception": "true"
        }
      }
    }
  ]
}
```

> **Note**
>
> Atualmente, essa chave de condição é suportada por um conjunto selecionado de serviços da AWS. O uso dessa chave com serviços não suportados pode levar a resultados de autorização não intencionais. Sempre especifique a chave de condição para serviços suportados em suas políticas.

> **Note**
>
> Quando Serviços da AWS fazem chamadas para outros Serviços da AWS em seu nome (chamadas entre serviços), um determinado contexto de autorização específico da rede é removido. Se sua política usa essa chave de condição com declarações `Deny`, as entidades principais do AWS service (Serviço da AWS) podem ser bloqueadas involuntariamente. Para permitir que os Serviços da AWS funcionem adequadamente enquanto mantém seus requisitos de segurança, exclua as entidades principais de serviço de suas declarações `Deny` adicionando a chave de condição `aws:PrincipalIsAWSService` com um valor de `false`.

aws:VpceOrgPaths

Use essa chave para comparar o caminho do AWS Organizations do endpoint da VPC do qual a solicitação foi feita com o caminho especificado na política. Essa chave de condição permite que você implemente controles de perímetro de rede no nível da unidade organizacional (UO), escalando automaticamente com o uso do seu endpoint da VPC à medida que você adiciona novos endpoints nas OUs especificadas.

- Disponibilidade: essa chave é incluída no contexto da solicitação quando uma solicitação é feita por meio de um endpoint da VPC e a conta do proprietário do endpoint da VPC é membro de uma organização. A chave não está incluída para solicitações feitas por meio de outros caminhos de rede ou quando a conta do proprietário do endpoint da VPC não faz parte de uma organização.

  Os serviços a seguir oferecem suporte a essa chave de condição:

  - AWS App Runner (prefixo: `apprunner`)
  - AWS Application Discovery Service (prefixo: `discovery`)
  - Amazon Athena (prefixo: `athena`)
  - AWS Cloud Map (prefixo: `servicediscovery`)
  - Amazon CloudWatch Application Insights (prefixo: `applicationinsights`)
  - AWS CloudFormation (prefixo: `cloudformation`)
  - Amazon Comprehend Medical (prefixo: `comprehendmedical`)
  - AWS Compute Optimizer (prefixo: `compute-optimizer`)
  - Amazon; Elastic Container Registry (prefixo: `ecr`)
  - Amazon Elastic Container Service (prefixo: `ecs`)
  - Amazon Kinesis Analytics (prefixo: `kinesisanalytics`)
  - Amazon Route 53 (prefixo: `route53`)
  - AWS DataSync (prefixo: `datasync`)
  - Amazon DynamoDB (prefixo: `dynamodb`)
  - Amazon Elastic Block Store (prefixo: `ebs`)
  - Agendador do Amazon EventBridge (prefixo: `scheduler`)
  - Amazon Data Firehose (prefixo: `firehose`)
  - Amazon GameLift (prefixo: `gamelift`)
  - AWS HealthImaging (prefixo: `medical-imaging`)

- AWS HealthLake (prefixo: `healthlake`)
- AWS HealthOmics (prefixo: `omics`)
- AWS Identity and Access Management(exceto para a ação `iam:PassRole`) (prefixo: `iam`)
- AWS IoT FleetWise (prefixo: `iotfleetwise`)
- AWS IoT Wireless (prefixo: `iotwireless`)
- AWS Key Management Service (prefixo: `kms`)
- AWS Lambda (prefixo: `lambda`)
- AWS Payment Cryptography (prefixo: `payment-cryptography`)
- Amazon Polly (prefixo: `polly`)
- Autoridade de certificação privada da AWS (prefixo: `acm-pca`)
- AWS Lixeira da (prefixo: `rbin`)
- Amazon Rekognition (prefixo: `rekognition`)
- Service Quotas (prefixo: `servicequotas`)
- AWS Secrets Manager (prefixo: `secretsmanager`)
- AWS Security Token Service (prefixo: `sts`)
- Amazon Simple Storage Service (prefixo: `s3`)
- AWS Shield (prefixo: `shield`)
- AWS Storage Gateway (prefixo: `storagegateway`)
- AWS Systems Manager Incident Manager Contatos da (prefixo: `ssm-contacts`)
- Amazon Textract (prefixo: `textract`)
- Amazon Transcribe (prefixo: `transcribe`)
- AWS Transfer Family (prefixo: `transfer`)

- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos
- Valores de exemplo – `o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/`

Como `aws:VpceOrgPaths` é uma chave de condição de vários valores, você deve usar os operadores de conjunto `ForAnyValue` ou `ForAllValues` com operadores de condição de string para essa chave. O seguinte exemplo de política de bucket do Amazon S3 permite acesso apenas quando as solicitações vem por meio de endpoints da VPC de propriedade da conta nas unidades organizacionais especificadas:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessFromSpecificOrgPaths",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/RoleName"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket1/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:VpceOrgPaths": [
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/*",
            "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-33333333/*"
          ]
        }
      }
    }
  ]
}
```

> **Note**
>
> Atualmente, essa chave de condição é suportada por um conjunto selecionado de serviços da AWS. O uso dessa chave com serviços não suportados pode levar a resultados de autorização não intencionais. Sempre especifique a chave de condição para serviços suportados em suas políticas.

Alguns serviços da AWS acessam seus recursos a partir de suas redes quando agem em seu nome. Se você usar esses serviços, precisará editar o exemplo de política acima para permitir que os serviços da AWS acessem seus recursos de fora da sua rede. Para obter mais informações sobre os padrões de acesso que precisam ser considerados ao aplicar controles de acesso com base na origem da solicitação, consulte Estabeleça barreiras de proteção para permissões usando perímetros de dados.

aws:VpcSourceIp

Use essa chave para comparar o endereço IP do qual uma solicitação foi feita com o endereço IP especificado na política. Em uma política, a chave será correspondente somente se a solicitação for proveniente do endereço IP especificado e passar por um VPC endpoint.

- Disponibilidade: essa chave será incluída no contexto da solicitação somente se a solicitação for feita usando um endpoint da VPC.
- Tipo de dados: Endereço IP
- Tipo de valor: valor único

Para obter mais informações, consulte Controlar o acesso aos endpoints da VPC usando políticas de endpoint no Guia do usuário da Amazon VPC. Assim como ao usar aws:SourceVpc, você deve usar aws:ViaAWSService ou aws:CalledVia para permitir solicitações feitas por Serviços da AWS usando sessões de acesso direto (FAS). Isso ocorre porque o IP da origem da solicitação inicial feita usando um endpoint da VPC não é preservado em solicitações de FAS.

> **Note**
>
> `aws:VpcSourceIp`O oferece suporte a endereços IP para IPv4 e IPv6 ou a um intervalo de endereços IP. Para obter uma lista de Serviços da AWS com suporte para IPv6, consulte Serviços da AWS que oferecem suporte a IPv6 no Guia do usuário da Amazon VPC. Sempre use a chave de condição `aws:VpcSourceIp` com as chaves de condição `aws:SourceVpc`, `aws:SourceVpce` ou `aws:SourceVpcArn`. Caso contrário, uma política pode permitir chamadas de API de uma VPC inesperada que usa o mesmo CIDR de IP ou CIDR sobreposto. Isso pode ocorrer porque os CIDRs de IP das duas VPCs não relacionadas podem ser os mesmos ou se sobrepor.

> **Note**
>
> Quando Serviços da AWS fazem chamadas para outros Serviços da AWS em seu nome (chamadas entre serviços), um determinado contexto de autorização específico da rede é removido. Se sua política usa essa chave de condição com declarações `Deny`, as entidades principais do AWS service (Serviço da AWS) podem ser bloqueadas involuntariamente. Para permitir que os Serviços da AWS funcionem adequadamente enquanto mantém seus

requisitos de segurança, exclua as entidades principais de serviço de suas declarações `Deny` adicionando a chave de condição `aws:PrincipalIsAWSService` com um valor de `false`.

Propriedades do recurso

Use as chaves de condição a seguir para comparar detalhes sobre o recurso que é o alvo da solicitação com as propriedades da entidade principal especificada na política.

aws:ResourceAccount

Use essa chave para comparar o ID da Conta da AWS do proprietário do recurso solicitado com a conta do recurso na política. Então, você pode permitir ou negar o acesso a esse recurso com base na conta proprietária do recurso.

- Availability (Disponibilidade): essa chave sempre é incluída no contexto da solicitação para a maioria das ações do serviço. As ações a seguir não são compatíveis com essa chave de condição:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store: todas as ações
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents` — O EventBridge `PutEvents` chama um barramento de eventos em outra conta, se esse barramento de eventos tiver sido configurado como um destino

do EventBridge entre contas antes de 2 de março de 2023. Para obter mais informações, consulte Conceder permissões para permitir eventos de outras contas da AWS no Guia do usuário do Amazon EventBridge.

- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- Amazon OpenSearch Service
  - `es:AcceptInboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
- AWS Security Hub CSPM
  - `securityhub:AcceptAdministratorInvitation`

- Tipo de dados: string
- Tipo de valor: valor único

> **Note**
>
> Para considerações adicionais sobre as ações sem suporte acima, consulte o repositório Exemplos de políticas de perímetro de dados.

Essa chave é igual ao ID da Conta da AWS para a conta com os recursos avaliados na solicitação.

Para a maioria dos recursos da sua conta, o ARN contém o ID da conta do proprietário desse recurso. Para determinados recursos, como buckets do Amazon S3, o ARN do recurso não inclui o ID da conta. Os dois exemplos a seguir mostram a diferença entre um recurso com um ID de conta no ARN e um ARN do Amazon S3 sem um ID de conta:

- `arn:aws:iam::123456789012:role/AWSExampleRole`: perfil do IAM criado e pertencente à conta 123456789012.
- `arn:aws:s3:::amzn-s3-demo-bucket2`: bucket do Amazon S3 criado e controlado dentro da conta da `111122223333`, não exibido no ARN.

Use o console, API ou CLI da AWS para encontrar todos os seus recursos e ARNs correspondentes.

Você redige uma política que nega permissões a recursos com base no ID da conta do proprietário do recurso. Por exemplo, a política baseada em identidade a seguir negará acesso ao recurso especificado se o recurso não pertencer à conta especificada.

Para usar esta política, substitua o texto do espaço reservado em itálico pelas informações da conta.

> **Important**
>
> Esta política não permite qualquer ação. Em vez disso, ela usa o efeito `Deny` que nega explicitamente o acesso a todas as ações não listadas na instrução que não pertencerem à conta listada. Use essa política em combinação com outras políticas que permitem acesso a recursos específicos.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:us-east-1:111122223333:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

Essa política nega acesso a todos os recursos para um serviço da AWS específico, a menos que a Conta da AWS especificada seja proprietária do recurso.

> **Note**
>
> Alguns Serviços da AWS exigem acesso aos recursos pertencentes à AWS que estão hospedados em outra Conta da AWS. O uso de `aws:ResourceAccount` em suas políticas baseadas em identidade podem afetar a capacidade da sua identidade de acessar esses recursos.

Certos serviços da AWS, como o AWS Data Exchange, dependem de acesso a recursos fora das suas Contas da AWS para operações normais. Se você usar o elemento `aws:ResourceAccount` em suas políticas, inclua instruções adicionais para criar isenções para esses serviços. O exemplo de política AWS: negar acesso aos recursos do Amazon S3 fora da sua conta, exceto o AWS Data Exchange demonstra como negar acesso com base na conta do recurso ao definir exceções para recursos de propriedade do serviço.

Use esse exemplo de política como modelo para criar suas próprias políticas personalizadas. Para obter mais informações, consulte a documentação do serviço.

aws:ResourceOrgPaths

Use essa chave para comparar o caminho do AWS Organizations para o recurso acessado com o caminho na política. Em uma política, essa chave de condição garante que o solicitante pertença a um membro da conta na raiz da organização ou unidades organizacionais (OUs) especificadas no AWS Organizations. Um caminho do AWS Organizations é uma representação de texto da estrutura de uma entidade do Organizations. Para obter mais informações sobre como usar e entender caminhos, consulte Compreender o caminho da entidade do AWS Organizations

- Availability (Disponibilidade): essa chave só é incluída no contexto da solicitação se a entidade principal for membro de uma organização. Essa chave de condição global não é compatível com as seguintes ações:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service

    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store: todas as ações
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents` — O EventBridge `PutEvents` chama um barramento de eventos em outra conta, se esse barramento de eventos tiver sido configurado como um destino do EventBridge entre contas antes de 2 de março de 2023. Para obter mais informações, consulte Conceder permissões para permitir eventos de outras contas da AWS no Guia do usuário do Amazon EventBridge.
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - Amazon OpenSearch Service
    - `es:AcceptInboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos

> **Note**
>
> Para considerações adicionais sobre as ações sem suporte acima, consulte o repositório Exemplos de políticas de perímetro de dados.

`aws:ResourceOrgPaths` é uma chave de condição de vários valores. Chaves de valores múltiplos podem ter vários valores no contexto da solicitação. Você deve usar os operadores de conjunto `ForAnyValue` ou `ForAllValues` com operadores de condição de string para essa chave. Para obter mais informações sobre chaves de condição de vários valores, consulte Operadores de conjunto para chaves de contexto de vários valores.

Por exemplo, a condição a seguir retorna `True` para recursos que pertencem à organização `o-a1b2c3d4e5`. Quando você inclui um caractere curinga, deve usar o operador de condição StringLike.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

A condição a seguir retorna `True` para recursos com o ID de UO `ou-ab12-11111111`. Ele corresponderá recursos pertencentes a contas anexadas à UO ou-ab12-11111111 ou a qualquer uma das UOs secundárias.

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

A condição a seguir retorna `True` para recursos pertencentes a contas anexadas diretamente à ID `ou-ab12-22222222` da UO, mas não às UOs secundárias. O exemplo a seguir usa o operador de condição StringEquals para especificar o requisito de correspondência exata para o ID da UO e não uma correspondência com curinga.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> Alguns Serviços da AWS exigem acesso aos recursos pertencentes à AWS que estão hospedados em outra Conta da AWS. O uso de `aws:ResourceOrgPaths` em suas políticas baseadas em identidade podem afetar a capacidade da sua identidade de acessar esses recursos.

Certos serviços da AWS, como o AWS Data Exchange, dependem de acesso a recursos fora das suas Contas da AWS para operações normais. Se você usar a chave `aws:ResourceOrgPaths` em suas políticas, inclua instruções adicionais para criar isenções para esses serviços. O exemplo de política AWS: negar acesso aos recursos do Amazon S3 fora da sua conta, exceto o AWS Data Exchange demonstra como negar acesso com base na conta do recurso ao definir exceções para recursos de propriedade do serviço. É possível criar uma política semelhante para restringir o acesso aos recursos dentro de uma unidade organizacional (UO) usando a chave `aws:ResourceOrgPaths`, levando em conta os recursos pertencentes ao serviço.

Use esse exemplo de política como modelo para criar suas próprias políticas personalizadas. Para obter mais informações, consulte a documentação do serviço.

aws:ResourceOrgID

Use essa chave para comparar o identificador da organização no AWS Organizations ao qual o solicitante pertence com o identificador especificado na política.

- Availability (Disponibilidade): essa chave só é incluída no contexto da solicitação se a entidade principal for membro de uma organização. Essa chave de condição global não é compatível com as seguintes ações:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - AWS Directory Service
    - `ds:AcceptSharedDirectory`
  - Amazon Elastic Block Store: todas as ações
  - Amazon EC2

    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents` — O EventBridge `PutEvents` chama um barramento de eventos em outra conta, se esse barramento de eventos tiver sido configurado como um destino do EventBridge entre contas antes de 2 de março de 2023. Para obter mais informações, consulte Conceder permissões para permitir eventos de outras contas da AWS no Guia do usuário do Amazon EventBridge.
  - Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - Amazon OpenSearch Service
    - `es:AcceptInboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
  - AWS Security Hub CSPM
    - `securityhub:AcceptAdministratorInvitation`
- Tipo de dados: string
- Tipo de valor: valor único

> **Note**
>
> Para considerações adicionais sobre as ações sem suporte acima, consulte o repositório Exemplos de políticas de perímetro de dados.

Essa chave global retorna o ID da organização do recurso para uma determinada solicitação. Ele permite que você crie regras que se aplicam a todos os recursos em uma organização que são especificados no elemento `Resource` de uma política baseada em identidade. É possível especificar o ID da organização no elemento de condição. Ao adicionar e remover contas, as políticas que incluem a chave `aws:ResourceOrgID` incluem automaticamente as contas corretas e não exigem atualização manual.

Por exemplo, a política a seguir impede que a entidade principal adicione objetos ao recurso `policy-genius-dev`, a menos que o recurso do Amazon S3 pertença à mesma organização que a entidade principal que faz a solicitação.

> **Important**
>
> Esta política não permite qualquer ação. Em vez disso, ela usa o efeito `Deny` que nega explicitamente o acesso a todas as ações não listadas na instrução que não pertencerem à conta listada. Use essa política em combinação com outras políticas que permitem acesso a recursos específicos.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": {
        "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
        "Effect": "Deny",
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::policy-genius-dev/*",
        "Condition": {
            "StringNotEquals": {
                "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
            }
        }
    }
}
```

> **Note**
>
> Alguns Serviços da AWS exigem acesso aos recursos pertencentes à AWS que estão hospedados em outra Conta da AWS. O uso de `aws:ResourceOrgID` em suas políticas baseadas em identidade podem afetar a capacidade da sua identidade de acessar esses recursos.

Certos serviços da AWS, como o AWS Data Exchange, dependem de acesso a recursos fora das suas Contas da AWS para operações normais. Se você usar a chave `aws:ResourceOrgID` em suas políticas, inclua instruções adicionais para criar isenções para esses serviços. O exemplo de política AWS: negar acesso aos recursos do Amazon S3 fora da sua conta, exceto o AWS Data Exchange demonstra como negar acesso com base na conta do recurso ao definir exceções para recursos de propriedade do serviço. É possível criar uma política semelhante para restringir o acesso aos recursos dentro da organização usando a chave `aws:ResourceOrgID`, levando em conta os recursos pertencentes ao serviço.

Use esse exemplo de política como modelo para criar suas próprias políticas personalizadas. Para obter mais informações, consulte a documentação do serviço.

No vídeo a seguir, saiba mais sobre como você pode usar a chave de condição `aws:ResourceOrgID` em uma política.

Garanta que identidades e redes só possam ser usadas para acessar recursos confiáveis.

aws:ResourceTag/tag-key

Use essa chave para comparar o par chave-valor da etiqueta especificado na política com o par chave-valor anexado ao recurso. Por exemplo, é possível exigir que o acesso a um recurso seja permitido somente se o recurso tiver a chave de tag `"Dept"` anexada com o valor `"Marketing"`. Para obter mais informações, consulte Controlar o acesso aos recursos do AWS.

- Availability (Disponibilidade): essa chave é incluída no contexto da solicitação quando o recurso solicitado já tem tags anexadas que criam um recurso com uma tag anexada. Esta chave é devolvida apenas para recursos que oferecem suporte à autorização com base em tags. Há uma chave de contexto para cada par de chave/valor de tag.
- Tipo de dados: String
- Tipo de valor: valor único

Essa chave de contexto é formatada `"aws:ResourceTag/tag-key":"tag-value"` em que *`tag-key`* e *`tag-value`* são uma chave de tag e um par de valores. As chaves de tag não diferenciam maiúsculas de minúsculas. Isso significa que, se você especificar `"aws:ResourceTag/TagKey1": "Value1"` no elemento de condição da política, a condição corresponderá a uma chave de tag de recurso chamada `TagKey1` ou `tagkey1`, mas não ambas. Os valores nesses pares chave/valor da tag diferenciam maiúsculas de minúsculas. Isso significa que, se você especificar `"aws:ResourceTag/TagKey1": "Production"` no elemento de condição da política, então a condição corresponderá a um valor de chave de tag de recurso com o nome `Production`, mas não a `production` ou `PRODUCTION`.

Para obter exemplos de como usar a chave `aws:ResourceTag` para controlar o acesso aos recursos do IAM, consulte Controlar o acesso aos recursos do AWS.

Para obter exemplos de uso da chave `aws:ResourceTag` para controlar o acesso a outros recursos AWS, consulte Controlar o acesso a recursos da AWS usando tags.

Para obter um tutorial sobre como usar a chave de condição `aws:ResourceTag` para o controle de acesso baseado em atributos (ABAC), consulte Tutorial do IAM: Definir permissões para acessar recursos da AWS com base em etiquetas.

Propriedades da solicitação

Use as chaves de condição a seguir para comparar detalhes sobre a solicitação em si e o conteúdo da solicitação e as propriedades da solicitação que você especificou na política.

aws:CalledVia

Use essa chave para comparar os serviços na política com os serviços que fizeram solicitações em nome da entidade principal do IAM (usuário ou perfil). Quando um principal faz uma solicitação a um serviço da AWS, esse serviço pode usar as credenciais do principal para fazer solicitações subsequentes a outros serviços. Quando a solicitação é feita usando sessões de acesso direto (FAS), essa chave é definida com o valor da entidade principal do serviço. A chave `aws:CalledVia` contém uma lista ordenada de cada serviço na cadeia que fez solicitações em nome do principal.

Para obter mais informações, consulte Sessões de acesso direto.

- Disponibilidade: essa chave está presente na solicitação quando um serviço que oferece suporte a `aws:CalledVia` usa as credenciais de uma entidade principal do IAM para fazer uma solicitação a outro serviço. Esta chave não estará presente se o serviço usar um perfil de serviço ou perfil

vinculado ao serviço para fazer uma chamada em nome da entidade principal. Esta chave também não está presente quando o principal faz a chamada diretamente.

- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos

Para usar a chave de condição `aws:CalledVia` em uma política, você deve fornecer as entidades principais do serviço para conceder ou negar solicitações do serviço da AWS. Por exemplo, é possível usar o AWS CloudFormation para ler e gravar de uma tabela do Amazon DynamoDB. O DynamoDB então usa a criptografia fornecida pelo AWS Key Management Service (AWS KMS).

Para conceder ou negar acesso quando qualquer serviço fizer uma solicitação usando as credenciais do principal, use a chave de condição `aws:ViaAWSService`. Essa chave de condição oferece suporte a todos os serviços AWS.

A chave `aws:CalledVia` é uma chave de vários valores. No entanto, você não pode aplicar uma ordem usando essa chave em uma condição. Ao usar o exemplo acima, o User 1 (Usuário 1) faz uma solicitação ao CloudFormation, que chama o DynamoDB, que chama o AWS KMS. São três solicitações distintas. A chamada final para o AWS KMS é realizada pelo Usuário 1 por meio do CloudFormation e, depois, via DynamoDB.
Exemplo de uso do aws:CalledVia.

Nesse caso, a chave `aws:CalledVia` no contexto de solicitação inclui `cloudformation.amazonaws.com` e `dynamodb.amazonaws.com`, nessa ordem. Se a sua única preocupação for que a chamada foi feita por meio do DynamoDB em algum lugar da cadeia de solicitações, você pode usar essa chave de condição em sua política.

Por exemplo, a política a seguir permite gerenciar a chave do AWS KMS chamada `my-example-key`, mas apenas se DynamoDB for um dos serviços solicitantes. O operador de condição `ForAnyValue:StringEquals` garante que o DynamoDB seja um dos serviços de chamada. Se o principal fizer a chamada diretamente para o AWS KMS, a condição retornará `false` e a solicitação não será permitida por esta política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": [
                        "dynamodb.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Se você quiser definir qual serviço fará a primeira ou última chamada na cadeia, use as chaves `aws:CalledViaFirst` e `aws:CalledViaLast`. Por exemplo, a política a seguir permite gerenciar a chave nomeada `my-example-key` no AWS KMS. Essas operações do AWS KMS são permitidas somente se várias solicitações tiverem sido incluídas na cadeia. A primeira solicitação deve ser feita pelo CloudFormation e a última pelo DynamoDB. Se outros serviços fizerem solicitações no meio da cadeia, a operação ainda será permitida.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
```

```
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

As chaves aws:CalledViaFirst e aws:CalledViaLast estão presentes na solicitação quando um serviço usa as credenciais de uma entidade principal do IAM para chamar outro serviço. Elas indicam o primeiro e último serviços que fizeram chamadas na cadeia de solicitações. Por exemplo, vamos supor que o CloudFormation chame outro serviço chamado `X Service`, que chama o DynamoDB que, por sua vez, chama o AWS KMS. A chamada final para o AWS KMS é realizada pelo `User 1` por meio do CloudFormation, depois pelo `X Service` e por fim pelo DynamoDB. Ele foi chamado pela primeira vez por meio do CloudFormation e a última chamada foi feita por meio do DynamoDB.
Exemplo de uso de aws:CalledViaFirst e aws:CalledViaLast.

aws:CalledViaFirst

Use esta chave para comparar os serviços na política com o primeiro serviço que fez uma solicitação em nome da entidade principal do IAM (usuário ou perfil). Para obter mais informações, consulte aws:CalledVia.

- Disponibilidade: essa chave está presente na solicitação quando um serviço usa as credenciais de uma entidade principal do IAM para fazer pelo menos uma outra solicitação a um serviço diferente. Esta chave não estará presente se o serviço usar um perfil de serviço ou perfil vinculado ao serviço para fazer uma chamada em nome da entidade principal. Esta chave também não está presente quando o principal faz a chamada diretamente.
- Tipo de dados: String
- Tipo de valor: valor único

aws:CalledViaLast

Use esta chave para comparar os serviços na política com o último serviço que fez uma solicitação em nome da entidade principal do IAM (usuário ou perfil). Para obter mais informações, consulte `aws:CalledVia`.

- Disponibilidade: essa chave está presente na solicitação quando um serviço usa as credenciais de uma entidade principal do IAM para fazer pelo menos uma outra solicitação a um serviço diferente. Esta chave não estará presente se o serviço usar um perfil de serviço ou perfil vinculado ao serviço para fazer uma chamada em nome da entidade principal. Esta chave também não está presente quando o principal faz a chamada diretamente.
- Tipo de dados: String
- Tipo de valor: valor único

aws:ViaAWSService

Use essa chave para verificar se um AWS service (Serviço da AWS) faz uma solicitação a outro serviço em seu nome usando sessões de acesso direto (FAS).

A chave de contexto da solicitação retorna `true` quando um serviço usa sessões de acesso direto para fazer uma solicitação em nome da entidade principal do IAM. A chave de contexto da solicitação também retorna `false` quando o principal faz a chamada diretamente.

- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: Booleano
- Tipo de valor: valor único

aws:ViaCustomerDomain

Use essa chave para verificar se uma entidade principal fez uma solicitação usando credenciais obtidas da Delegação de acesso temporário (TAD). Quando um parceiro ou fornecedor obtém credenciais temporárias por meio de uma troca de tokens TAD, definimos essa chave como o identificador da conta do parceiro solicitante.

Use essa chave de condição nas políticas do Identity and Access Management (IAM) da AWS para definir o escopo dos controles de acesso especificamente às sessões estabelecidas pela delegação de acesso temporário, sem afetar suas próprias sessões. Por exemplo, você pode usar essa chave para revogar somente sessões baseadas em delegação, preservando o acesso direto ao console.

- Disponibilidade — Essa chave está presente no contexto da solicitação quando uma entidade principal usa credenciais obtidas por meio de uma troca de tokens da TAD.
- Tipo de dados: string
- Tipo de valor: valor único

O exemplo a seguir mostra uma política de negação para sessões obtidas por meio da delegação de acesso temporário de uma conta de parceiro específica. A política se aplica somente às sessões emitidas antes de um horário especificado. Este exemplo usa um curinga (*) para `Action` e `Resource` na instrução `Deny`, que é o padrão para revogar todo o acesso de uma sessão específica. Para revogar somente ações específicas, substitua o caractere curinga pela lista de ações apropriada.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "DateLessThan": {
                    "aws:TokenIssueTime": "2026-07-30T01:00:37.819Z"
                },
                "StringEquals": {
                    "aws:ViaCustomerDomain": "123456789012"
                }
            }
        }
    ]
}
```

aws:CalledViaAWSMCP

Use essa chave para comparar os serviços na política com os serviços do AWS MCP que fizeram solicitações em nome da entidade principal do IAM (usuário ou perfil). Quando um principal faz uma solicitação a um serviço da AWS, esse serviço usa as credenciais do principal para fazer solicitações subsequentes a outros serviços. Quando a solicitação é feita usando um serviço do AWS MCP, essa chave é definida com o valor da entidade principal do serviço. A chave `aws:CalledViaAWSMCP`

contém o nome da entidade principal do serviço do MCP que fez solicitações em nome da entidade principal.

- Disponibilidade: essa chave está presente na solicitação quando um serviço do AWS MCP usa as credenciais de uma entidade principal do IAM para fazer uma solicitação a um serviço da AWS. Esta chave também não está presente quando o principal faz a chamada diretamente.
- Tipo de dados: String
- Tipo de valor: valor único

Você pode usar essa chave de condição para conceder ou negar acesso com base em qual servidor MCP específico iniciou a solicitação. Por exemplo, a política a seguir nega operações de exclusão sensíveis quando elas são iniciadas por meio de um servidor MCP específico:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaSpecificMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaAWSMCP": "aws-mcp.amazonaws.com"
                }
            }
        }
    ]
}
```

aws:ViaAWSMCPService

Use essa chave para verificar se um serviço do AWS MCP faz uma solicitação a outro serviço da AWS em seu nome usando sessões de acesso direto (FAS). A chave de contexto da solicitação retorna `true` quando um serviço do AWS MCP encaminha uma solicitação para um serviço da AWS

em nome da entidade principal do IAM. A chave de contexto da solicitação também retorna `false` quando o principal faz a chamada diretamente.

- Disponibilidade: essa chave será incluída no contexto da solicitação quando um servidor AWS MCP fizer uma solicitação a um serviço da AWS downstream em nome de uma entidade principal do IAM.
- Tipo de dados: Booleano
- Tipo de valor: valor único

É possível usar essa chave para restringir ações específicas quando elas chegam por meio de servidores MCP. Por exemplo, a política a seguir nega operações de exclusão sensíveis quando elas são iniciadas por meio de qualquer servidor AWS MCP:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenySensitiveActionsViaMCP",
            "Effect": "Deny",
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "dynamodb:DeleteTable"
            ],
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "aws:ViaAWSMCPService": "true"
                }
            }
        }
    ]
}
```

aws:CurrentTime

Use essa chave para comparar a data e a hora da solicitação com a data e a hora especificadas na política. Para visualizar um exemplo de política que usa essa chave de condição, consulte AWS: Permite o acesso com base na data e hora.

- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: Data
- Tipo de valor: valor único

aws:EpochTime

Use essa chave para comparar a data e a hora da solicitação em horário Unix ou epoch com o valor especificado na política. Essa chave também aceita o número de segundos desde 1º de janeiro de 1970.

- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: Data, Numérico
- Tipo de valor: valor único

aws:referer

Use essa chave para comparar quem indicou a solicitação no navegador cliente com o indicador especificado na política. O valor de contexto da solicitação `aws:referer` é fornecido pelo chamador em um cabeçalho HTTP. O cabeçalho `Referer` é incluído em uma solicitação de navegador da Web quando você seleciona um link em uma página da Web. O cabeçalho `Referer` contém o URL da página da Web onde o link foi selecionado.

- Disponibilidade: essa chave será incluída no contexto da solicitação apenas se a solicitação para o recurso da AWS for invocada vinculando um URL de página da Web no navegador. Essa chave não está incluída para solicitações programáticas porque não usa um link do navegador para acessar o recurso AWS.
- Tipo de dados: String
- Tipo de valor: valor único

Por exemplo, você pode acessar um objeto do Amazon S3 diretamente, usando um URL ou a invocação direta da API. Para obter mais informações, consulte Operações de API do Amazon S3 diretamente usando um navegador da Web. Quando você acessa um objeto do Amazon S3 de um URL que existe em uma página da Web, o URL da página da Web de origem é usado em `aws:referer`. Quando você acessa um objeto do Amazon S3 digitando o URL no seu navegador, o `aws:referer` não está presente. Quando você invoca a API diretamente, `aws:referer` também não está presente. É possível usar a chave da condição `aws:referer` em uma política para permitir

solicitações feitas de um referencial específico, como um link em uma página da Web no domínio da sua empresa.

> **Warning**
>
> Essa chave deve ser usada com cuidado. É perigoso incluir um valor de cabeçalho do indicador conhecido publicamente. Partes não autorizadas podem usar navegadores personalizados ou modificados para fornecer qualquer valor de `aws:referer` que escolherem. Como resultado, `aws:referer` não deve ser usado para impedir que terceiros não autorizados façam solicitações diretas da AWS. Ele é oferecido apenas para permitir que os clientes impeçam que seu conteúdo digital, como o conteúdo armazenado no Amazon S3, seja indicado em sites de terceiros não autorizados.

aws:RequestedRegion

Use essa chave para comparar a região da AWS que foi chamada na solicitação com a região especificada na política. É possível usar essa chave de condição global para controlar quais regiões podem ser solicitadas. Para visualizar as regiões da AWS de cada serviço, consulte Cotas e endpoints de serviço na Referência geral da Amazon Web Services.

- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: String
- Tipo de valor: valor único

Alguns serviços globais, como o IAM, têm um único endpoint. Como esse endpoint está fisicamente localizado na região Leste dos EUA (Norte da Virgínia), as chamadas do IAM sempre são feitas para a região us-east-1. Por exemplo, se você criar uma política que negue acesso a todos os serviços se a região solicitada não for us-west-2, as chamadas do IAM sempre falharão. Para ver um exemplo de como resolver isso, consulte NotAction com Deny.

> **Note**
>
> A chave de condição `aws:RequestedRegion` permite que você controle qual endpoint de um serviço é invocado, mas não abrange o impacto da operação. Alguns serviços têm impactos entre regiões.
> Por exemplo, o Amazon S3 tem operações de API que abrangem todas as regiões.

- É possível invocar `s3:PutBucketReplication` em uma região (que é afetada pela chave de condição `aws:RequestedRegion`), mas outras regiões são afetadas com base nas definições da configuração das replicações.
- É possível invocar `s3:CreateBucket` para criar um bucket em outra região e usar a chave de condição `s3:LocationConstraint` para controlar as regiões aplicáveis.

É possível usar essa chave de contexto para limitar o acesso aos serviços da AWS em um determinado conjunto de regiões. Por exemplo, a política a seguir permite que um usuário visualize todas as instâncias do Amazon EC2 no Console de gerenciamento da AWS. No entanto, eles só podem realizar alterações às instâncias na Irlanda (eu-west-1), Londres (eu-west-2) ou Paris (eu-west-3).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
```

```
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

aws:RequestTag/tag-key

Use essa chave para comparar o par de chave/valor da tag que foi passado na solicitação com o par de tags especificado na política. Por exemplo, é possível verificar se a solicitação inclui a chave de tag `"Dept"` e se ela tem o valor `"Accounting"`. Para obter mais informações, consulte Controlar o acesso durante solicitações do AWS.

- Availability (Disponibilidade): essa chave é incluída no contexto da solicitação quando os pares de chave-valor são passadas na solicitação. Quando várias tags forem passadas na solicitação, haverá uma chave de contexto para cada par de chave/valor de tag.
- Tipo de dados: String
- Tipo de valor: valor único

Essa chave de contexto é formatada `"aws:RequestTag/`*`tag-key`*`":"`*`tag-value`*`"` em que *`tag-key`* e *`tag-value`* são uma chave de tag e um par de valores. As chaves de tag não diferenciam maiúsculas de minúsculas. Isso significa que, se você especificar `"aws:RequestTag/TagKey1": "Value1"` no elemento de condição da sua política, a condição corresponderá a uma chave de etiqueta de solicitação chamada `TagKey1` ou `tagkey1`, mas não ambas. Os valores nesses pares chave/valor da tag diferenciam maiúsculas de minúsculas. Isso significa que, se você especificar `"aws:RequestTag/TagKey1": "Production"` no elemento de condição da política, então a

condição corresponderá a um valor de chave de tag de solicitação com o nome `Production`, mas não a `production` ou `PRODUCTION`.

Este exemplo mostra que, embora a chave tenha um único valor, você ainda poderá usar vários pares de chave-valor em uma solicitação se as chaves forem diferentes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2::111122223333:instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

aws:TagKeys

Use essa chave para comparar as chaves de tag em uma solicitação com as chaves especificadas na política. Ao usar políticas para controlar o acesso usando etiquetas, recomendamos o uso da chave de condição `aws:TagKeys` para definir quais chaves de etiquetas serão permitidas. Para obter mais informações e políticas de exemplo, consulte the section called “Controlar o acesso com base em chaves de tag”.

- Availability (Disponibilidade): essa chave será incluída no contexto da solicitação se a operação for compatível com a passagem de tags no recurso.
- Tipo de dados: String (lista)

- Tipo de valor: valores múltiplos

Essa chave de contexto é formatada como `"aws:TagKeys":"tag-key"`, onde *`tag-key`* é uma lista de chaves de tags sem valores (por exemplo, `["Dept","Cost-Center"]`).

Como é possível incluir vários pares de chave/valor de tag em uma solicitação, o conteúdo da solicitação pode ser uma solicitação de vários valores. Nesse caso, você deve usar os operadores de conjunto `ForAllValues` ou `ForAnyValue`. Para obter mais informações, consulte Operadores de conjunto para chaves de contexto de vários valores.

Alguns serviços são compatíveis com o uso de tags com operações de recurso, como criar, modificar ou excluir um recurso. Para permitir o uso de tags e operações como uma única chamada, você deve criar uma política que inclui as ações de uso de tags e de modificação de recursos. Em seguida, você pode usar a chave de condição `aws:TagKeys` para impor o uso de chaves de tags específicas na solicitação. Por exemplo, para limitar as etiquetas quando alguém cria um snapshot do Amazon EC2, você deve incluir a ação de criação `ec2:CreateSnapshot` e a ação de etiquetamento `ec2:CreateTags` na política. Para visualizar uma política para esse cenário que use `aws:TagKeys`, consulte Criar um snapshot com tags no Guia do usuário do Amazon EC2.

aws:SecureTransport

Use essa chave para verificar se a solicitação foi enviada via TLS. O contexto da solicitação retorna `true` ou `false`. Em uma política, você poderá permitir ações específicas somente se a solicitação for enviada via TLS.

- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: Booleano
- Tipo de valor: valor único

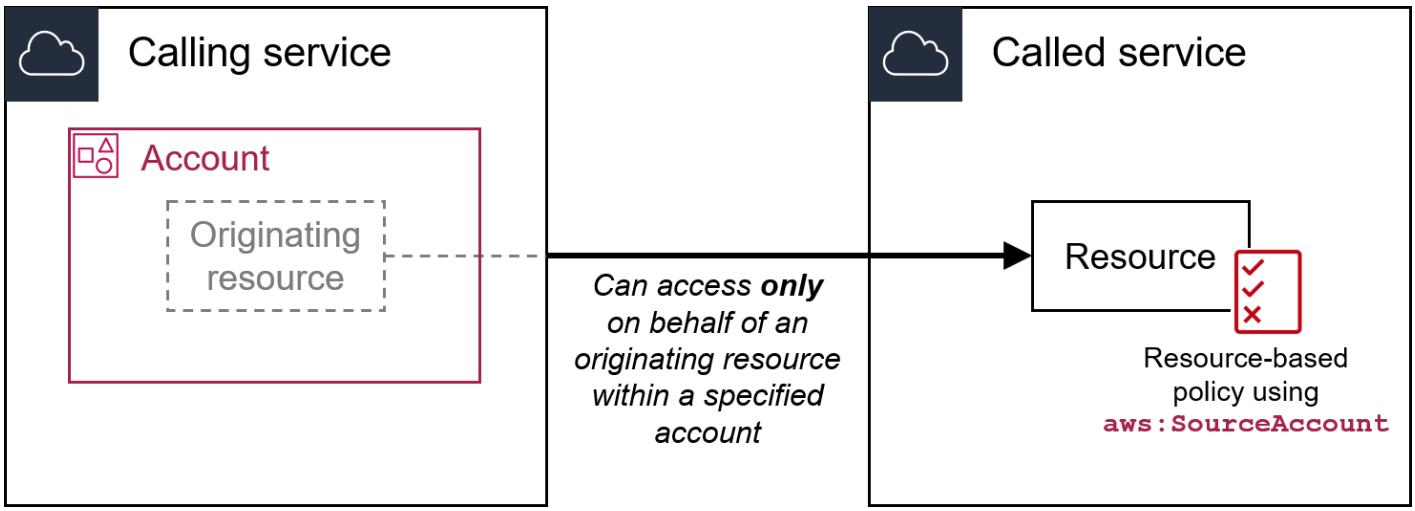
aws:SourceAccount

Use essa chave para comparar o ID da conta do recurso que faz uma solicitação de serviço a serviço com o ID da conta especificado na política, mas apenas quando a solicitação for feita pela entidade principal de um serviço da AWS.

- Disponibilidade: essa chave só é incluída no contexto da solicitação quando a chamada para o recurso é feita diretamente pela entidade principal de um serviço da AWS em nome de um

recurso para o qual a configuração acionou a solicitação de serviço a serviço. O serviço que faz a chamada passa o ID da conta do recurso original para o serviço chamado.

> **Note**
>
> Essa chave fornece um mecanismo uniforme para impor um controle de representante confuso entre serviços em todos os Serviços da AWS. No entanto, nem todas as integrações de serviços exigem o uso dessa chave de condição global. Consulte a documentação dos Serviços da AWS usados para obter mais informações sobre os mecanismos específicos dos serviços para mitigar os riscos de representante confuso entre serviços.



- Tipo de dados: string
- Tipo de valor: valor único

É possível usar essa chave de condição para ajudar a certificar-se de que um serviço que faça uma chamada possa acessar seu recurso somente quando a solicitação for originada de uma conta específica. Por exemplo, é possível anexar a política de controle de recursos (RCP) a seguir para negar solicitações de entidades de serviço para buckets do Amazon S3, a menos que tenham sido disparadas por um recurso na conta especificada. Essa política aplica o controle somente a solicitações das entidades principais de serviço (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) que tenham a chave `aws:SourceAccount` presente (`"Null": {"aws:SourceAccount": "false"}`), de modo que as integrações de serviços que não exijam o uso dessa chave e as chamadas de suas entidades principais não sejam

afetadas. Se a chave `aws:SourceAccount` estiver presente no contexto da solicitação, a condição `Null` será avaliada como `true`, fazendo com que a chave `aws:SourceAccount` seja aplicada.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceAccount": "111122223333"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

Em políticas baseadas em recursos em que a entidade principal é uma entidade principal de AWS service (Serviço da AWS), use a chave para limitar as permissões concedidas ao serviço. Por exemplo, quando um bucket do Amazon S3 é configurado para enviar notificações a um tópico do Amazon SNS, o serviço do Amazon S3 invoca a operação de API `sns:Publish` para todos os eventos configurados. Na política de tópico que permite a operação `sns:Publish`, defina o valor da chave de condição como o ID da conta do bucket do Amazon S3.

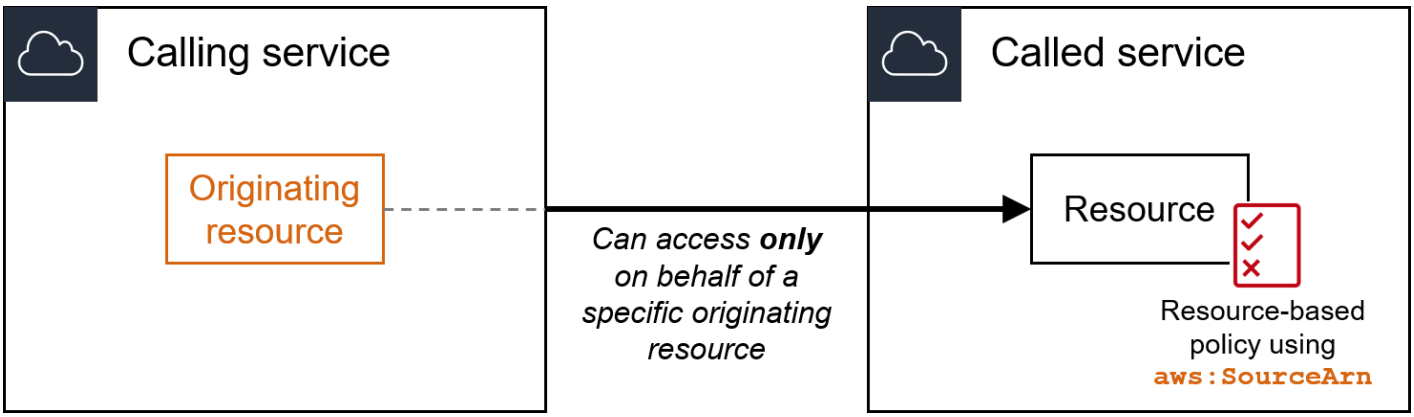
aws:SourceArn

Use essa chave para comparar o nome do recurso da Amazon (ARN) do recurso que faz uma solicitação de serviço a serviço com o ARN especificado na política, mas só quando a solicitação for feita pela entidade principal do serviço da AWS. Quando o ARN da fonte inclui o ID da conta, não é necessário usar a `aws:SourceAccount` com o `aws:SourceArn`.

Essa chave não funciona com o ARN do principal que está fazendo a solicitação. Em seu lugar, use `aws:PrincipalArn`.

- Disponibilidade: essa chave só é incluída no contexto da solicitação quando a chamada para o recurso é feita diretamente pela entidade principal de um serviço da AWS em nome de um recurso para o qual a configuração acionou a solicitação de serviço a serviço. O serviço que faz a chamada passa o ARN do recurso original para o serviço chamado.

> **Note**
>
> Essa chave fornece um mecanismo uniforme para impor um controle de representante confuso entre serviços em todos os Serviços da AWS. No entanto, nem todas as integrações de serviços exigem o uso dessa chave de condição global. Consulte a documentação dos Serviços da AWS usados para obter mais informações sobre os mecanismos específicos dos serviços para mitigar os riscos de representante confuso entre serviços.



- Tipo de dados: ARN

  A AWS recomenda utilizar operadores ARN em vez de operadores string ao comparar ARNs.
- Tipo de valor: valor único

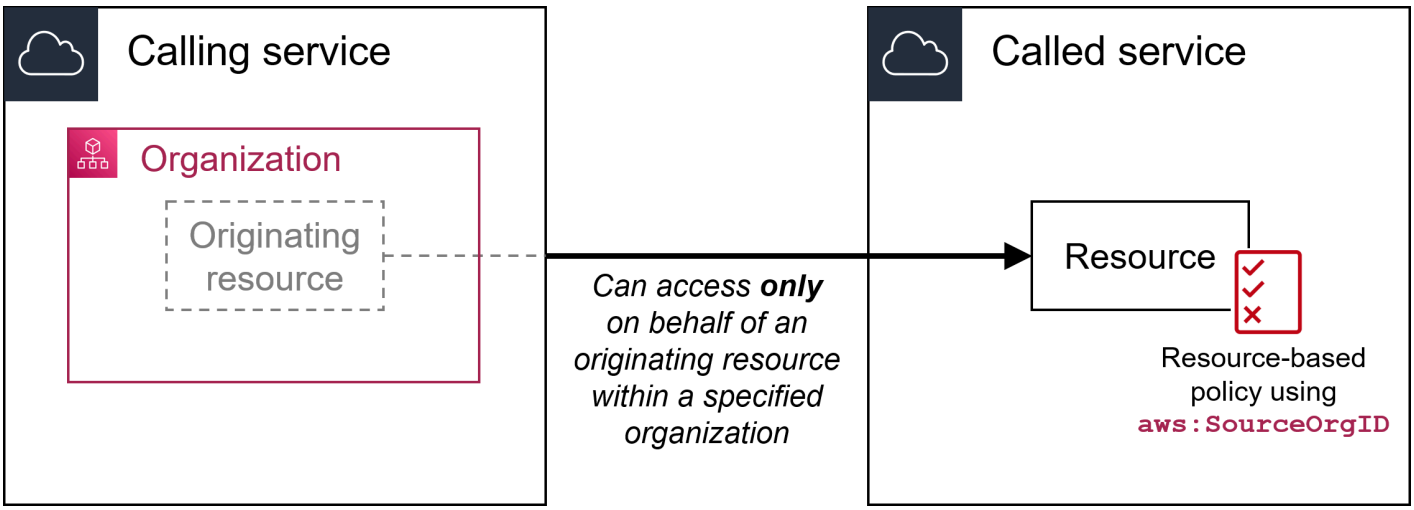
É possível usar essa chave de condição para ajudar a certificar-se de que um serviço que faça uma chamada possa acessar seu recurso somente quando a solicitação for originada de um recurso específico. Ao usar uma política baseada em recursos com uma entidade principal de AWS service (Serviço da AWS) como a `Principal`, defina o valor dessa chave de condição como o ARN do recurso ao qual você deseja restringir o acesso. Por exemplo, quando um bucket do Amazon S3 é configurado para enviar notificações a um tópico do Amazon SNS, o serviço do Amazon S3 invoca a operação de API `sns:Publish` para todos os eventos configurados. Na política de tópico que permite a operação `sns:Publish`, defina o valor da chave de condição como o ARN do bucket do Amazon S3. Para obter recomendações sobre quando usar essa chave de condição em políticas baseadas em recursos, consulte a documentação dos Serviços da AWS que você está usando.

aws:SourceOrgID

Use essa chave para comparar o ID da organização do recurso que faz uma solicitação de serviço a serviço com o ID da organização especificado na política, mas apenas quando a solicitação for feita pela entidade principal de um serviço da AWS. Quando você adiciona e remove contas de uma organização no AWS Organizations, as políticas que incluem a chave `aws:SourceOrgID` incluem automaticamente as contas corretas e você não precisa atualizar as políticas manualmente.

- Disponibilidade: essa chave só é incluída no contexto da solicitação quando a chamada ao recurso é feita diretamente pela entidade principal de um serviço da AWS em nome de um recurso que pertence a uma conta que é membro de uma organização. O serviço que faz a chamada passa o ID da organização do recurso original para o serviço chamado.

> **Note**
>
> Essa chave fornece um mecanismo uniforme para impor um controle de representante confuso entre serviços em todos os Serviços da AWS. No entanto, nem todas as integrações de serviços exigem o uso dessa chave de condição global. Consulte a documentação dos Serviços da AWS usados para obter mais informações sobre os mecanismos específicos dos serviços para mitigar os riscos de representante confuso entre serviços.



- Tipo de dados: string
- Tipo de valor: valor único

É possível usar essa chave de condição para ajudar a certificar-se de que um serviço que faça uma chamada possa acessar seu recurso somente quando a solicitação for originada de uma organização específica. Por exemplo, é possível anexar a política de controle de recursos (RCP) a seguir para negar solicitações de entidades principais para buckets do Amazon S3, a menos que tenham sido disparadas por um recurso na organização da AWS especificada. Essa política aplica o controle somente a solicitações das entidades principais de serviço (`"Bool": {"aws:PrincipalIsAWSService": "true"}`) que tenham a chave `aws:SourceAccount` presente (`"Null": {"aws:SourceAccount": "false"}`), de modo que as integrações de serviços que não exijam o uso da chave e as chamadas de suas entidades principais não sejam afetadas. Se a chave `aws:SourceAccount` estiver presente no contexto da solicitação, a condição `Null` será avaliada como `true`, fazendo com que a chave `aws:SourceOrgID` seja aplicada. Usamos `aws:SourceAccount` em vez de `aws:SourceOrgID` no operador de condição `Null` para que o controle ainda se seja aplicado caso a solicitação se origine de uma conta que não pertença a uma organização.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "RCPEnforceConfusedDeputyProtection",
      "Effect": "Deny",
      "Principal": "*",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceOrgID": "o-xxxxxxxxxx"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
    }
  ]
}
```

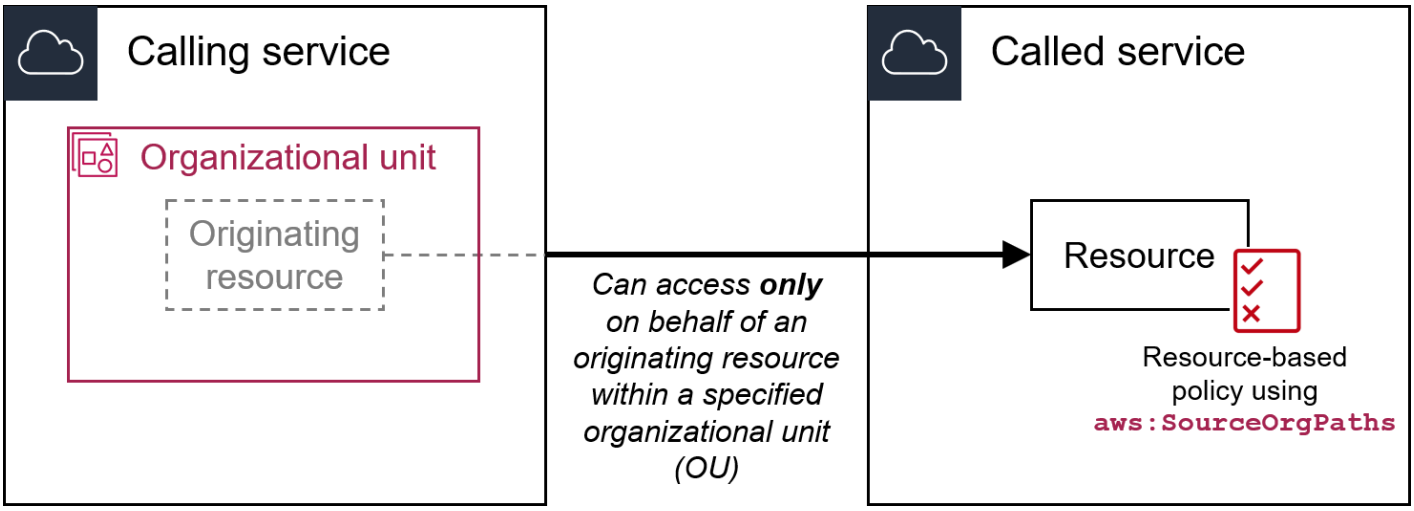
aws:SourceOrgPaths

Use essa chave para comparar o caminho do AWS Organizations do recurso que faz uma solicitação de serviço a serviço com o ID da organização especificado na política, mas apenas quando a solicitação for feita pela entidade principal de um serviço da AWS. Um caminho AWS Organizations é uma representação de texto da estrutura de uma entidade do AWS Organizations. Para obter mais informações sobre como usar e entender os caminhos, consulte Entender o caminho da entidade do AWS Organizations.

- Disponibilidade: essa chave só é incluída no contexto da solicitação quando a chamada ao recurso é feita diretamente pela entidade principal de um serviço da AWS em nome de um recurso que pertence a uma conta que é membro de uma organização. O serviço que faz a chamada passa o caminho da organização do recurso original para o serviço chamado.

> **Note**
>
> Essa chave fornece um mecanismo uniforme para impor um controle de representante confuso entre serviços em todos os Serviços da AWS. No entanto, nem todas as

integrações de serviços exigem o uso dessa chave de condição global. Consulte a documentação dos Serviços da AWS usados para obter mais informações sobre os mecanismos específicos dos serviços para mitigar os riscos de representante confuso entre serviços.



- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos

Use essa chave de condição para ajudar a certificar-se de que um serviço que faça uma chamada possa acessar seu recurso somente quando a solicitação for originada de uma unidade organizacional (UO) específica no AWS Organizations.

Da mesma forma que em `aws:SourceOrgID`, para ajudar a evitar impactos nas integrações de serviços que não exijam o uso dessa chave, use o operador de condição `Null` com a chave de condição `aws:SourceAccount` para que o controle ainda seja aplicado se a solicitação se originar de uma conta que não pertença a uma organização.

```
{
      "Condition": {
        "ForAllValues:StringNotLikeIfExists": {
            "aws:SourceOrgPaths": "o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-
ab12-22222222/"
        },
        "Null": {
          "aws:SourceAccount": "false"
        },
```

```
        "Bool": {
          "aws:PrincipalIsAWSService": "true"
        }
      }
}
```

`aws:SourceOrgPaths` é uma chave de condição de vários valores. Chaves de valores múltiplos podem ter vários valores no contexto da solicitação. Você deve usar os operadores de conjunto `ForAnyValue` ou `ForAllValues` com operadores de condição de string para essa chave. Para obter mais informações sobre chaves de condição de vários valores, consulte Operadores de conjunto para chaves de contexto de vários valores.

aws:UserAgent

Use essa chave para comparar o aplicativo cliente do solicitante com o aplicativo especificado na política.

- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: String
- Tipo de valor: valor único

> ⚠ Warning
>
> Essa chave deve ser usada com cuidado. Como o valor `aws:UserAgent` é fornecido pelo chamador em um cabeçalho HTTP, partes não autorizadas podem usar navegadores personalizados ou modificados para fornecer qualquer valor `aws:UserAgent` que escolherem. Como resultado, `aws:UserAgent` não deve ser usado para impedir que terceiros não autorizados façam solicitações diretas da AWS. É possível usá-lo para permitir apenas aplicações cliente específicas, e somente depois de testar sua política.

aws:IsMcpServiceAction

Use esta chave para verificar se a ação que está sendo autorizada é uma ação de serviço MCP. Essa chave não se refere às ações executadas pelo serviço MCP para outros serviços da AWS.

- Disponibilidade: esta chave está incluída no contexto da solicitação e será definida como True somente quando o serviço MCP autorizar uma ação de serviço MCP.

- Tipo de dados: Booleano
- Tipo de valor: valor único

Outras chaves de condição nos serviços

O AWS STS oferece suporte a chaves de condição de federação baseadas em SAML e chaves de condição entre serviços para federação OIDC. Essas chaves estão disponíveis quando um usuário que foi federado usando OIDC ou SAML executa operações da AWS em outros serviços.

Chaves de contexto de condição do IAM e do AWS STS

É possível usar o elemento `Condition` em uma política JSON para testar o valor das chaves que estão incluídas no contexto de solicitação de todas as solicitações da AWS. Essas chaves fornecem informações sobre a solicitação em si, ou os recursos referenciados pela solicitação. É possível verificar que as chaves foram especificadas antes de permitir a ação solicitada pelo usuário. Isso oferece a você controle granular sobre quando as instruções de sua política JSON correspondem ou não a uma solicitação recebida. Para obter informações sobre como usar o elemento `Condition` em uma política JSON, consulte Elementos de política JSON do IAM: Condition.

Este tópico descreve as chaves definidas e fornecidas pelo serviço IAM (com um prefixo `iam:`) e o serviço AWS Security Token Service (AWS STS) (com um prefixo `sts:`). Vários outros serviços da AWS também fornecem chaves específicas de serviços que são relevantes para as ações e os recursos definidos por esse serviço. Para obter mais informações, consulte Ações, recursos e chaves de condição de serviços da AWS. A documentação de um serviço que dá suporte a chaves de condição, muitas vezes, tem informações adicionais. Por exemplo, para informações sobre chaves que você pode usar em políticas para recursos do Amazon S3, consulte Chaves de política do Amazon S3 no Guia do usuário do Amazon Simple Storage Service.

Tópicos

- Teclas disponíveis para o IAM
- Chaves disponíveis para federação OIDC da AWS
- Chaves disponíveis para federação do AWS STS com base em SAML
- Chaves de contexto de federação do AWS STS baseadas em SAML entre serviços
- Chaves disponíveis do AWS STS

Teclas disponíveis para o IAM

É possível usar as seguintes chaves de condição em políticas que controlam o acesso aos recursos do IAM:

iam:AccountPropertyNamespaces

Funciona com operadores de string.

Especifica os namespaces da propriedade da conta que estão sendo lidos ou modificados na solicitação. As propriedades da conta são organizadas em namespaces (por exemplo, `RoleManager`). Ao chamar `PutAccountProperties` ou `GetAccountProperties`, use essa chave de condição para restringir quais namespaces de propriedade uma entidade principal pode acessar.

iam:AssociatedResourceArn

Funciona com operadores de nome de recurso da Amazon (ARN).

Especifica o ARN do recurso ao qual essa função será associada no serviço de destino. O recurso geralmente pertence ao serviço ao qual o principal está transmitindo a função. Às vezes, o recurso pode pertencer a um terceiro serviço. Por exemplo, você pode passar uma função para o Amazon EC2 Auto Scaling que seja usada em uma instância do Amazon EC2. Nesse caso, a condição corresponderia ao ARN da instância do Amazon EC2.

Essa chave de condição se aplica somente à ação PassRole em uma política. Ela não pode ser usada para limitar nenhuma outra ação.

> **Important**
>
> Ao usar a condição `iam:AssociatedResourceArn` em uma política para restringir a ação PassRole, considerações especiais se aplicarão se a política tiver a intenção de definir o acesso à ação AddRoleToInstanceProfile. Nesse caso, você não pode especificar uma ID de região ou instância no ARN da instância do EC2. O valor do ARN deve ser `arn:aws:ec2:*:CallerAccountId:instance/*`. O uso de qualquer outro valor de ARN pode levar a resultados de avaliação inesperados.

Use essa chave de condição em uma política baseada em identidade para permitir que uma entidade transmita um perfil, mas somente se esse perfil estiver associado ao recurso

especificado. Por exemplo, você pode permitir que um usuário do IAM ou um perfil transmita qualquer perfil para o serviço do Amazon EC2 a ser usado com instâncias na Conta da AWS. O usuário ou a função do IAM não terá permissão para passar funções para outros serviços.

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": "ec2.amazonaws.com"
        },
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:*:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> Os serviços da AWS que oferecem suporte ao IAM:PassedToService também oferecem suporte a esta chave de condição.

iam:AWSServiceName

Funciona com operadores de string.

Especifica o serviço da AWS para o qual essa função é anexada.

Essa chave de condição é suportada pela operação da API `CreateServiceLinkedRole`.

> **Tip**
>
> Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte AWSServiços da que funcionam com o IAM e procure os serviços que têm Sim na coluna Função vinculada ao serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Neste exemplo, você permite que uma entidade do crie um perfil vinculado ao serviço usando a operação da API `CreateServiceLinkedRole`, se o nome do serviço for access-analyzer.amazonaws.com.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
     }]
 }
```

iam:FIDO-certification

Funciona com operadores de string.

Verifica o nível de certificação de FIDO do dispositivo MFA no momento do registro de uma chave de segurança FIDO. A certificação do dispositivo é obtida do Serviço de metadados (MDS) da FIDO Alliance. Se o status ou o nível de certificação de sua chave de segurança FIDO mudar, ele não será atualizado, a menos que o dispositivo tenha o registrado cancelado e seja registrado novamente para buscar as informações de certificação atualizadas.

Valores possíveis de L1, L1plus, L2, L2plus, L3, L3plus

Neste exemplo, você registra uma chave de segurança e recupera a certificação FIDO Nível 1 plus para seu dispositivo.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-2-certification

Funciona com operadores de string.

Verifica o nível de certificação de validação FIPS-140-2 do dispositivo MFA no momento do registro de uma chave de segurança FIDO. A certificação do dispositivo é obtida do Serviço de metadados (MDS) da FIDO Alliance. Se o status ou o nível de certificação de sua chave de segurança FIDO mudar, ele não será atualizado, a menos que o dispositivo tenha o registrado cancelado e seja registrado novamente para buscar as informações de certificação atualizadas.

Valores possíveis de L1, L2, L3, L4

Neste exemplo, você registra uma chave de segurança e recupera a certificação FIPS-140-2 Nível 2 do seu dispositivo.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
```

```
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-3-certification

Funciona com operadores de string.

Verifica o nível de certificação de validação FIPS-140-3 do dispositivo MFA no momento do registro de uma chave de segurança FIDO. A certificação do dispositivo é obtida do Serviço de metadados (MDS) da FIDO Alliance. Se o status ou o nível de certificação de sua chave de segurança FIDO mudar, ele não será atualizado, a menos que o dispositivo tenha o registrado cancelado e seja registrado novamente para buscar as informações de certificação atualizadas.

Valores possíveis de L1, L2, L3, L4

Neste exemplo, você registra uma chave de segurança e recupera a certificação FIPS-140-3 Nível 3 para seu dispositivo.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
```

```
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
            }
        }
    ]

}
```

iam:OrganizationsPolicyId

Funciona com operadores de string.

Verifica se a política com o ID do AWS Organizations especificado corresponde à política usada na solicitação. Para ver um exemplo de política do IAM que use essa chave de condição, consulte IAM: visualizar serviço acessado por último para uma política do AWS Organizations.

iam:PassedToService

Funciona com operadores de string.

Especifica o serviço principal do serviço para o qual uma função pode ser transmitida. Essa chave de condição se aplica somente à ação PassRole em uma política. Ela não pode ser usada para limitar nenhuma outra ação.

Ao usar essa chave de condição em uma política, especifique o serviço usando um principal de serviço. Um serviço principal é o nome de um serviço que pode ser especificado no elemento `Principal` de uma política. Este é o formato normal: `SERVICE_NAME_URL.amazonaws.com`.

É possível usar `iam:PassedToService` para restringir os usuários, para que eles possam passar funções apenas para serviços específicos. Por exemplo, um usuário pode criar uma função de serviço que confia no CloudWatch para gravar dados de log em um bucket do Amazon S3 em seu nome. Em seguida, o usuário deve anexar uma política de permissões e uma política de confiança para a nova função de serviço. Nesse caso, a política de confiança deve especificar `cloudwatch.amazonaws.com` no elemento `Principal`. Para visualizar uma política que permite ao usuário passar a função para o CloudWatch, consulte IAM: Passar uma função do IAM para um produto da AWS específico.

Ao usar essa chave de condição, você pode certificar-se de que os usuários criem perfis de serviço apenas para os serviços que você especificar. Por exemplo, se um usuário com a política anterior tentar criar uma função de serviço para o Amazon EC2, a operação falhará. A falha ocorre porque o usuário não tem permissão para passar a função para o Amazon EC2.

Às vezes, você passa uma função para um serviço que, em seguida, passa a função para outro serviço. `iam:PassedToService` inclui apenas o serviço final que assume a função, não o serviço intermediário que passa a função.

> **Note**
>
> Alguns serviços não são compatíveis com essa chave de condição.

iam:PermissionsBoundary

Funciona com operadores de nome de recurso da Amazon (ARN).

Verifica se a política especificada está anexada como limite de permissões no recurso da entidade principal do IAM. Para obter mais informações, consulte Limites de permissões para entidades do IAM

iam:PolicyARN

Funciona com operadores de nome de recurso da Amazon (ARN).

Verifica o nome de recurso da Amazon (ARN) de uma política gerenciada em solicitações que envolvem uma política gerenciada. Para obter mais informações, consulte Controle de acesso a políticas.

iam:RoleTemplateARN

Funciona com operadores de nome de recurso da Amazon (ARN).

Especifica o ARN do modelo de perfil usado para criar ou gerenciar um perfil. Quando um serviço da AWS adquire um perfil de um modelo de perfil, o ARN do modelo de perfil é incluído como uma chave de contexto nas verificações de autorização para `CreateRole`, `TagRole`, `PutRolePolicy`, `AttachRolePolicy`, `PutRolePermissionsBoundary` e `GetRole`.

iam:RegisterSecurityKey

Funciona com operadores de string.

Verifica o estado atual da ativação do dispositivo MFA.

Valores possíveis de `Create` ou `Activate`.

Neste exemplo, você registra uma chave de segurança e recupera a certificação FIPS-140-3 Nível 1 para seu dispositivo.

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

 }
```

iam:ResourceTag/***key-name***

Funciona com operadores de string.

Verifica se a tag anexada ao recurso de identidade (usuário ou função) corresponde ao nome e ao valor da chave especificada.

> **Note**
>
> O IAM e o AWS STS são compatíveis com a chave de condição do IAM `iam:ResourceTag` e a chave de condição global `aws:ResourceTag`.

É possível adicionar atributos personalizados aos recursos do IAM na forma de um par de chave-valor. Para obter mais informações sobre etiquetas de recursos do IAM, consulte the section called “Tags para recursos do IAM”. É possível usar `ResourceTag` para controlar o acesso aos recursos da AWS, incluindo recursos do IAM. No entanto, como o IAM não oferece suporte a etiquetas para grupos, você não pode usar etiquetas para controlar o acesso a grupos.

Este exemplo mostra como você pode criar uma política baseada em identidade que permite excluir usuários com a tag **`status=terminated`**. Para usar esta política, substitua o *`texto do espaço reservado em itálico`* na política de exemplo por suas próprias informações. Em seguida, siga as instruções em criar uma política ou editar uma política.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status":
 "terminated"}}
    }]
}
```

iam:ServiceSpecificCredentialAgeDays

Funciona com operadores numéricos.

Essa chave de condição limita a criação de credenciais específicas do serviço com base em suas configurações de expiração. Ele permite que você controle a idade máxima, em dias, das credenciais específicas do serviço que podem ser criadas.

O intervalo válido para dias é de 1 a 36600 (mínimo 1 dia, máximo 36600 dias).

Essa chave de condição é suportada pela operação da API `CreateServiceSpecificCredential`.

Neste exemplo, você permite que um usuário crie credenciais específicas para o serviço Amazon Bedrock somente se elas expirarem em 90 dias.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                },
                "NumericLessThanEquals": {
                    "iam:ServiceSpecificCredentialAgeDays": "90"
                }
            }
        }
    ]
}
```

iam:ServiceSpecificCredentialServiceName

Funciona com operadores de string.

Especifica quais serviços da AWS podem ser usados ao gerenciar credenciais específicas do serviço. Essa chave de condição permite restringir quais serviços da AWS são permitidos ao gerenciar credenciais específicas do serviço.

Esta chave de condição é suportada pelas seguintes operações de API:

- CreateServiceSpecificCredential
- DeleteServiceSpecificCredential
- ResetServiceSpecificCredential
- UpdateServiceSpecificCredential

Os serviços a seguir são compatíveis com credenciais específicas do serviço com a formatação exata do valor:

- bedrock.amazonaws.com
- cassandra.amazonaws.com
- codecommit.amazonaws.com

Neste exemplo, você permite que um usuário crie credenciais específicas para a operação de API CreateServiceSpecificCredential , apenas para o serviço do Amazon Bedrock.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceSpecificCredential",
            "Resource": "arn:aws:iam::111122223333:user/username",
            "Condition": {
                "StringEquals": {
                    "iam:ServiceSpecificCredentialServiceName":
 "bedrock.amazonaws.com"
                }
            }
        }
    ]
}
```

iam:DelegationDuration

Funciona com operadores numéricos.

Filtra o acesso com base na duração do acesso temporário solicitado em uma solicitação de delegação.

Os provedores de produtos podem usar essa chave de condição para controlar a duração máxima que eles permitem em solicitações de delegação enviadas aos clientes. A duração é especificada em segundos e determina por quanto tempo as credenciais temporárias permanecem válidas após o cliente liberar o token de troca. Isso ajuda os provedores de produtos a aplicar políticas internas sobre os limites de duração do acesso com base em seus casos de uso.

Essa chave de condição é compatível com a operação da API `CreateDelegationRequest`.

Neste exemplo, você permitirá criar solicitações de delegação somente se a duração solicitada for de 7.200 segundos (2 horas) ou menos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "NumericLessThanEquals": {
                    "iam:DelegationDuration": "7200"
                }
            }
        }
    ]
}
```

iam:NotificationChannel

Funciona com operadores de nome de recurso da Amazon (ARN).

Filtra o acesso com base no ARN do tópico do Amazon SNS especificado para receber notificações de solicitação de delegação.

Os provedores de produtos podem usar essa chave de condição para restringir quais tópicos do SNS podem ser usados para notificações de solicitação de delegação na chamada da API CreateDelegationRequest. Os provedores de produtos devem especificar um tópico do SNS para receber notificações de mudança de estado e tokens de troca. Isso garante que as notificações sejam enviadas somente aos canais aprovados dentro da organização do provedor do produto.

Essa chave de condição é compatível com a operação da API `CreateDelegationRequest`.

Neste exemplo, você permitirá criar solicitações de delegação somente se elas usarem um tópico específico do SNS para notificações.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:NotificationChannel": "arn:aws:sns:us-
east-1:123456789012:delegation-notifications"
                }
            }
        }
    ]
}
```

iam:TemplateArn

Funciona com operadores de nome de recurso da Amazon (ARN).

Filtra o acesso com base no ARN do modelo de política usado para definir permissões em uma solicitação de delegação.

Os provedores de produtos podem usar essa chave de condição para controlar quais modelos de política podem ser usados na chamada da API CreateDelegationRequest. Os modelos de política definem as permissões temporárias que os provedores de produtos solicitam nas contas dos clientes. Isso permite que os provedores de produtos restrinjam quais modelos de política registrados podem ser usados ao criar solicitações de delegação.

Essa chave de condição é compatível com a operação da API `CreateDelegationRequest`.

Neste exemplo, você permitirá criar solicitações de delegação somente se elas usarem um modelo de política de um domínio de parceiro específico.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": "iam:CreateDelegationRequest",
            "Resource": "*",
            "Condition": {
                "ArnLike": {
                    "iam:TemplateArn": "arn:aws:iam::*:delegation-template/
partner_*"
                }
            }
        }
    ]
}
```

iam:DelegationRequestOwner

Funciona com operadores de nome de recurso da Amazon (ARN).

Filtra o acesso com base na entidade principal ou na identidade da AWS proprietária da solicitação de delegação.

Os clientes podem usar essa chave de condição para controlar quem pode realizar ações em solicitações de delegação com base na propriedade. O proprietário de uma solicitação de delegação é a entidade principal ou a identidade da AWS na conta do cliente que iniciou ou recebeu a solicitação de delegação.

Esta chave de condição é suportada pelas seguintes operações de API:

- `GetDelegationRequest`
- `AcceptDelegationRequest`
- `RejectDelegationRequest`
- `SendDelegationToken`
- `ListDelegationRequests`
- `UpdateDelegationRequest`

Neste exemplo, você permitirá que os usuários gerenciem somente solicitações de delegação de sua propriedade.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetDelegationRequest",
                "iam:AcceptDelegationRequest",
                "iam:RejectDelegationRequest",
                "iam:SendDelegationToken",
                "iam:UpdateDelegationRequest",
                "iam:ListDelegationRequests"
            ],
            "Resource": "*",
            "Condition": {
                "ArnEquals": {
                    "iam:DelegationRequestOwner": "${aws:PrincipalArn}"
                }
            }
        }
    ]
}
```

Chaves disponíveis para federação OIDC da AWS

É possível usar a federação OIDC para fornecer credenciais de segurança temporárias a usuários que foram autenticados por meio de um provedor de identidade (IdP) compatível com OpenID Connect em um provedor de identidade OpenID Connect (OIDC) do IAM em sua conta da AWS. Exemplos desses provedores incluem GitHub, Amazon Cognito, Login da Amazon, e Google. É possível usar tokens de identidade e de acesso do seu próprio IdP, bem como tokens de contas de serviço concedidos a workloads do Amazon Elastic Kubernetes Service.

É possível usar chaves de contexto de condição do AWS OIDC para criar políticas que limitam o acesso de entidades principais federadas a recursos associados a um provedor, aplicação ou usuário específico. Essas chaves são normalmente usadas na política de confiança de uma função. Defina as chaves de condição usando o nome do provedor do OIDC (`token.actions.githubusercontent.com`), seguido pela declaração (`:aud`): **`token.actions.githubusercontent.com:aud`**.

Algumas chaves de condição da federação do OIDC podem ser usadas na sessão de perfil para autorizar o acesso a recursos. Se o valor for Sim na coluna Disponível na sessão, você poderá usar essas chaves de condição nas políticas para definir o que os usuários têm permissão para acessar em outros serviços da AWS. Quando uma declaração não está disponível na sessão, a chave de

contexto de condição do OIDC só pode ser usada em uma política de confiança de função para a autenticação inicial AssumeRoleWithWebIdentity.

Selecione seu IdP para ver como as declarações do seu IdP são mapeadas para as chaves de contexto de condição do IAM na AWS. Você pode encontrar mais informações sobre chaves para o GitHub e o Google na guia Padrão.

Default

O padrão lista as declarações padrão do OIDC e como elas são mapeadas para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP. Use esse mapeamento se seu IdP não estiver listado nas opções da guia.

Os fluxos de trabalho do GitHub Actions e o Google são alguns exemplos de IDPs que usam a implementação padrão em seu token de ID JWT do OIDC.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
amr	amr	Sim
aud	azp Se nenhum valor for definido para `azp`, a chave de condição `aud` será mapeada para a declaração `aud`.	Sim
email	email	Não
oaud	aud	Não
sub	sub	Sim

Para obter mais informações sobre o uso de chaves de contexto de condição com o GitHub, consulte Configurar uma função para o provedor de identidades OIDC GitHub. Para obter mais informações sobre o Google e sobre os campos `aud` e `azp`, consulte o guia Google Identity Platform OpenID Connect.

amr

Funciona com operadores de string. A chave tem vários valores, o que significa que você a testa em uma política usando operadores do conjunto de condições.

Exemplo: `token.actions.githubusercontent.com:amr`

A Referência de métodos de autenticação inclui informações de login sobre o usuário. A chave pode conter os seguintes valores:

- Se o usuário não estiver autenticado, a chave conterá apenas `unauthenticated`.
- Se o usuário estiver autenticado, a chave conterá o valor `authenticated` e o nome do provedor de login usado na chamada (`accounts.google.com`).

aud

Funciona com operadores de string.

Examples:

- `accounts.google.com:aud`
- `token.actions.githubusercontent.com:aud`

Use a chave de condição `aud` para verificar se o público corresponde àquele especificado na política. É possível usar a chave de pub com a chave de ass para o mesmo provedor de identidade.

Essa chave de condição é definida a partir dos seguintes campos de token:

- `aud` para IDs de cliente do Google do OAuth 2.0 do aplicativo, quando o campo `azp` não estiver definido. Quando o campo `azp` estiver definido, o campo `aud` corresponderá à chave de condição `accounts.google.com:oaud`.
- `azp` quando o campo `azp` estiver definido. Isso pode acontecer com aplicativos híbridos nos quais um aplicativo web e um aplicativo Android têm um ID de cliente do Google do OAuth 2.0 diferente, mas compartilham o mesmo projeto de APIs do Google.

Quando você escreve uma política usando a chave de condição `accounts.google.com:aud`, é necessário saber se o aplicativo é um aplicativo híbrido que define o campo `azp`.

`azp`Campo não definido

O exemplo de política a seguir funciona para aplicativos não híbridos que não definem o campo `azp`. Nesse caso, o valor do campo `aud` do token de ID do Google

corresponde aos valores da chave de condição `accounts.google.com:aud` e `accounts.google.com:oaud`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

`azp`Campo definido

O exemplo de política a seguir funciona para aplicativos híbridos que definem o campo `azp`. Nesse caso, o valor do campo `aud` do token de ID do Google corresponde apenas ao valor da chave de condição `accounts.google.com:oaud`. O valor do campo `azp` corresponde ao valor da chave de condição `accounts.google.com:aud`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
```

```
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

email

Funciona com operadores de string.

Exemplo: `accounts.google.com:email`

Essa chave de condição valida o endereço de e-mail do usuário. O valor dessa declaração pode não ser exclusivo dessa conta e pode mudar com o tempo, portanto, você não deve usar esse valor como identificador primário para verificar seu registro de usuário.

oaud

Funciona com operadores de string.

Exemplo: `accounts.google.com:oaud`

Essa chave especificará o outro público (`aud`) ao qual esse token de ID se destina. Ela deve ser um dos IDs de cliente do OAuth 2.0 do aplicativo.

sub

Funciona com operadores de string.

Examples:

- `accounts.google.com:sub`
- token.actions.githubusercontent.com:sub

Use essas chaves para verificar se o assunto corresponde àquele especificado na política. É possível usar a chave `sub` com a chave `aud` para o mesmo provedor de identidade.

Na política de confiança de perfil a seguir, a chave de condição `sub` limita o perfil à ramificação do GitHub chamada `demo`.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub": "repo:org-name/repo-
name:ref:refs/heads/demo"
        }
      }
    }
  ]
}
```

Amazon Cognito

Essa guia explica como o Amazon Cognito mapeia as declarações do OIDC para as chaves de contexto de condição AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

Para perfis usados pelo Amazon Cognito, as chaves são definidas usando `cognito-identity.amazonaws.com` seguido pela declaração.

Para obter mais informações sobre mapeamento de declaração do banco de identidades, consulte Mapeamentos padrão do provedor no Guia do desenvolvedor do Amazon Cognito. Para obter mais informações sobre mapeamento de declaração do grupo de usuários, consulte Como usar token de ID no Guia do desenvolvedor do Amazon Cognito.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
amr	amr	Sim
aud	aud	Sim

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
oaud	aud	Não
sub	sub	Sim

amr

Funciona com operadores de string. A chave tem vários valores, o que significa que você a testa em uma política usando operadores do conjunto de condições.

Exemplo – `cognito-identity.amazonaws.com:amr`

A Referência de métodos de autenticação inclui informações de login sobre o usuário. A chave pode conter os seguintes valores:

- Se o usuário não estiver autenticado, a chave conterá apenas `unauthenticated`.
- Se o usuário estiver autenticado, a chave conterá o valor `authenticated` e o nome do provedor de login usado na chamada (`cognito-identity.amazonaws.com`).

Por exemplo, a seguinte condição na política de confiança para um perfil do Amazon Cognito testa se o usuário não está autenticado.

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

Funciona com operadores de string.

Exemplo – `cognito-identity.amazonaws.com:aud`

O cliente de aplicação do grupo de usuários que autenticou o usuário. O Amazon Cognito renderiza o mesmo valor na reivindicação `client_id` do token de acesso.

oaud

Funciona com operadores de string.

Exemplo – `cognito-identity.amazonaws.com:oaud`

O cliente de aplicação do grupo de usuários que autenticou o usuário. O Amazon Cognito renderiza o mesmo valor na reivindicação `client_id` do token de acesso.

sub

Funciona com operadores de string.

Exemplo – `cognito-identity.amazonaws.com:sub`

Um identificador exclusivo (UUID), ou assunto, do usuário autenticado. O nome de usuário pode não ser exclusivo em seu grupo de usuários. A subreivindicação é a melhor maneira de identificar um usuário específico. É possível usar a chave `sub` com a chave `aud` para o mesmo provedor de identidade.

```
"Condition": {
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
```

Login with Amazon

Essa guia explica como o Login da Amazon mapeia as declarações do OIDC para as chaves de contexto de condição AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
app_id	ID da aplicação	Sim
sub	ID de usuário	Sim
user_id	ID de usuário	Sim

app_id

Funciona com operadores de string.

Exemplo – `www.amazon.com:app_id`

Essa chave especifica o contexto do público que corresponde ao campo `aud` usado por outros provedores de identidade.

sub

Funciona com operadores de string.

Exemplo – `www.amazon.com:sub`

Essa chave verifica se o ID de usuário corresponde àquele especificado na política. É possível usar a chave `sub` com a chave `aud` para o mesmo provedor de identidade.

user_id

Funciona com operadores de string.

Exemplo – `www.amazon.com:user_id`

Essa chave especifica o contexto do público que corresponde ao campo `aud` usado por outros provedores de identidade. É possível usar a chave `user_id` com a chave `id` do mesmo provedor de identidade.

Facebook

Essa guia explica como o Facebook mapeia as declarações do OIDC para as chaves de contexto de condição AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
app_id	ID da aplicação	Sim
id	id	Sim

app_id

Funciona com operadores de string.

Exemplo – `graph.facebook.com:app_id`

Essa chave verificará se o contexto do público corresponde ao campo `aud` usado por outros provedores de identidade.

ID de

Funciona com operadores de string.

Exemplo – `graph.facebook.com:id`

Essa chave verificou se o ID da aplicação (ou site) corresponde àquele especificado na política.

GitHub

Essa guia explica como o GitHub Actions mapeia as declarações OIDC para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
actor	actor	Não
actor_id	actor_id	Não
job_workflow_ref	job_workflow_ref	Não
repositório	repositório	Não
repository_id	repository_id	Não
repository_owner_id	repository_owner_id	Não
fluxo de trabalho	fluxo de trabalho	Não

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
ref	ref	Não
ambiente	ambiente	Não
enterprise_id	enterprise_id	Não

actor

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:actor`

Essa chave identifica a conta pessoal que iniciou a execução do fluxo de trabalho. Use-a para restringir o acesso a agentes específicos.

actor_id

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:actor_id`

Essa chave verifica o ID da conta pessoal que iniciou a execução do fluxo de trabalho. Os IDs de agentes são gerados pelo GitHub e são imutáveis.

job_workflow_ref

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:job_workflow_ref`

Essa chave contém o caminho de referência para o fluxo de trabalho reutilizável de trabalhos que usam esse tipo de fluxo de trabalho. Use-a para restringir o acesso a fluxos de trabalho específicos e certificar-se de que somente os fluxos de trabalho aprovados possam assumir perfis.

Repositório do

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:repository`

Essa chave identifica o repositório no qual o fluxo de trabalho é executado. Use-a para limitar o acesso a repositórios do GitHub específicos.

repository_id

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:repository_id`

Essa chave verifica o ID do repositório no qual o fluxo de trabalho é executado. Os IDs de repositórios são imutáveis e não serão alterados mesmo que o repositório seja renomeado.

repository_owner_id

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:repository_owner_id`

Esta chave verifica o ID da organização na qual o repositório está armazenado. O ID do proprietário do repositório é um identificador único e estável que não muda.

Fluxo de trabalho do

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:workflow`

Essa chave contém o nome do fluxo de trabalho. Use-a para restringir o acesso a fluxos de trabalho específicos nos repositórios.

ref

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:ref`

Essa chave identifica a git ref (ramificação ou tag) que acionou a execução do fluxo de trabalho. Use-a para limitar o acesso com base em ramificações específicas, como permitir somente as ramificações `main` ou `production`.

Ambiente do

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:environment`

Essa chave contém o nome do ambiente usado pelo trabalho. Use-a para implementar controles de acesso baseados no ambiente, como permissões separadas para os ambientes de desenvolvimento, preparação e produção.

> **Note**
>
> Se a declaração de ambiente for incluída na política de confiança, um ambiente deverá ser configurado e fornecido no fluxo de trabalho do GitHub.

enterprise_id

Funciona com operadores de string.

Exemplo – `token.actions.githubusercontent.com:enterprise_id`

Essa chave verifica o ID do repositório no qual fluxo de trabalho é executado. Use-a para certificar-se de que o acesso seja limitado aos repositórios na sua organização do GitHub Enterprise.

> **Use identificadores imutáveis, não nomes**
>
> No GitHub, o repositório, a organização e os nomes de usuário podem ser alterados. Um nome que é liberado ao renomear ou excluir pode ser reivindicado por uma conta diferente. As políticas que dependem exclusivamente de reivindicações mutáveis baseadas em nomes (como `repository` ou `actor`) podem conceder acesso a identidades não intencionais. Para ajudar a se proteger contra isso, use os identificadores imutáveis que o GitHub fornece, como `repository_id`, `repository_owner_id` ou `actor_id`, em suas políticas de confiança de perfil ou políticas de controle de recursos.

O exemplo de política de confiança a seguir usa declarações personalizadas no token OIDC do GitHub para limitar o acesso a um perfil.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
```

```
        "Principal": {
            "Federated": "arn:aws:iam::111122223333:oidc-provider/
token.actions.githubusercontent.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "StringEquals": {
                "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
                "token.actions.githubusercontent.com:repository_owner_id": "123456",
                "token.actions.githubusercontent.com:repository_id": "1296269",
                "token.actions.githubusercontent.com:actor_id": "1",
                "token.actions.githubusercontent.com:ref": "refs/heads/main",
                "token.actions.githubusercontent.com:enterprise_id": "345"
              }
           }
        }
    ]
}
```

Google

Essa guia explica como o Google mapeia as declarações OIDC para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
google/organization_number	google:organization_number	Não

google/organization_number

Funciona com operadores numéricos.

Exemplo – `accounts.google.com:google/organization_number`

Essa chave confirma que um token representa uma identidade do Google que pertence a uma organização específica do Google Cloud ou do Google Workspace. Use-a para restringir o acesso a usuários de organizações específicas, garantindo que somente as identidades da sua organização possam assumir o perfil.

O exemplo de política de confiança a seguir usa a declaração `google/organization_number` para limitar o acesso a um perfil.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "accounts.google.com"
         },
         "Action": "sts:AssumeRoleWithWebIdentity",
         "Condition": {
            "NumericEquals": {
                "accounts.google.com:google/organization_number": "123456"
               }
           }
        }
    ]
}
```

CircleCI

Essa guia explica como o CircleCI mapeia as declarações OIDC para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
oidc.circleci.com/project-id	oidc.circleci.com/project-id	Não

oidc.circleci.com/project-id

Funciona com operadores de string.

Exemplo – `circleci-issuer-url:oidc.circleci.com/project-id`

Essa chave identifica o projeto do CircleCI no qual o trabalho é executado. Seu valor é uma string que contém um UUID que identifica exclusivamente o projeto do CircleCI. Use-a para restringir o acesso a projetos do CircleCI específicos.

O exemplo de política de confiança a seguir usa a declaração `oidc.circleci.com/project-id` para limitar o acesso a um perfil.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/oidc.circleci.com/
org/12345"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.circleci.com/org/12345:aud": "sts.amazonaws.com",
          "oidc.circleci.com/org/12345:oidc.circleci.com/project-id": "76543210-
ba98-fedc-3210-edcba0987654"
        }
      }
    }
  ]
}
```

Oracle Cloud Infrastructure (OCI)

Essa guia explica como o Oracle Cloud Infrastructure mapeia as declarações OIDC para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
rpst_id	rpst_id	Não

rpst_id

Funciona com operadores de string.

Exemplo – `oci-issuer-url:rpst_id`

Essa chave identifica de modo exclusivo a entidade principal do recurso no OCI. Use-a para restringir o acesso a entidades principais do OCI específicas. O rpst_id (ID de token de sessão da entidade principal do recurso) fornece um identificador estável para autenticação baseada no recurso do OCI.

O exemplo de política de confiança a seguir usa a declaração `rpst_id` para limitar o acesso a um perfil.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/idcs-
abc123ef5678901234abcd.identity.oraclecloud.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:aud":
 "sts.amazonaws.com",
          "idcs-abc123ef5678901234abcd.identity.oraclecloud.com:rpst_id": "your-
rpst-id"
        }
      }
    }
  ]
}
```

GitLab.com

Essa guia explica como o CI/CD do GitLab.com mapeia as reivindicações OIDC para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

> **Note**
>
> Essas chaves de condição personalizadas somente são aceitas quando a instância SaaS multilocatário do Gitlab.com é usada como provedor de identidade OIDC. Não é possível

usar essas chaves de condição com as instâncias GitLab Dedicated ou GitLab Self-Managed.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
namespace_id	namespace_id	Não
project_id	project_id	Não
user_id	user_id	Não
user_login	user_login	Não
user_email	user_email	Não
user_access_level	user_access_level	Não
ref_protected	ref_protected	Não
pipeline_source	pipeline_source	Não
runner_environment	runner_environment	Não

namespace_id

Funciona com operadores de string.

Exemplo – `gitlab.com:namespace_id`

Essa chave identifica o ID do namespace (grupo) do projeto que executa a tarefa de CI/CD. O ID do namespace é um identificador estável e exclusivo que não é alterado se o grupo é renomeado.

project_id

Funciona com operadores de string.

Exemplo – `gitlab.com:project_id`

Essa chave identifica o ID do projeto que executa a tarefa de CI/CD. O ID do projeto é um identificador estável e exclusivo que não é alterado se o projeto é renomeado ou movido.

user_id

Funciona com operadores de string.

Exemplo – `gitlab.com:user_id`

Essa chave identifica o ID do usuário que executa a tarefa de CI/CD. O ID do usuário é um identificador exclusivo e estável.

user_login

Funciona com operadores de string.

Exemplo – `gitlab.com:user_login`

Essa chave identifica o nome de usuário do usuário que executa a tarefa de CI/CD. Os nomes de usuário podem ser alterados; para obter um identificador estável, use `user_id`.

user_email

Funciona com operadores de string.

Exemplo – `gitlab.com:user_email`

Essa chave identifica o e-mail do usuário que executa a tarefa de CI/CD. O valor dessa declaração pode ser alterado posteriormente; para obter um identificador estável, use `user_id`.

user_access_level

Funciona com operadores de string.

Exemplo – `gitlab.com:user_access_level`

Essa chave identifica o nível de acesso do usuário no projeto (por exemplo, `maintainer`, `developer`, `owner`).

ref_protected

Funciona com operadores de string.

Exemplo – `gitlab.com:ref_protected`

Essa chave indica se a referência do Git que acionou a tarefa está protegida. O valor é `true` se estiver protegida; caso contrário, `false`.

pipeline_source

Funciona com operadores de string.

Exemplo – `gitlab.com:pipeline_source`

Essa chave identifica a fonte que acionou o pipeline (por exemplo, `push`, `web`, `schedule`, `api`, `merge_request_event`).

runner_environment

Funciona com operadores de string.

Exemplo – `gitlab.com:runner_environment`

Esta chave identifica o tipo de executor usado pelo trabalho de CI/CD.

> **Important**
>
> No SaaS do Gitlab.com, caminhos de grupos ou projetos excluídos podiam ser recriados por usuários diferentes, o que poderia fazer com que políticas que dependessem exclusivamente de declarações baseadas em caminhos (por exemplo, `sub`) concedessem acesso a identidades não pretendidas. Em 1º de junho de 2026, o GitLab implantou uma atualização no lado da plataforma que impede que novos tokens de ID de CI sejam emitidos para caminhos de projetos que até então emitiam tokens, reduzindo esse risco no futuro. Como camada adicional de proteção, a AWS e o GitLab recomendam incluir condições em identificadores estáveis e exclusivos, como `namespace_id` ou `project_id` ou, em seu perfil, políticas de confiança ou políticas de controle de recursos.

O exemplo de política de confiança a seguir usa a reivindicação `namespace_id` em conjunto com `sub` para limitar o acesso ao projeto e à ramificação específicos do GitLab.com.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::123456789012:oidc-provider/gitlab.com"
```

```
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "gitlab.com:sub": "project_path:mygroup/
myproject:ref_type:branch:ref:main",
          "gitlab.com:namespace_id": "12345"
        }
      }
    }
  ]
}
```

Buildkite

Esta guia explica como o CircleCI mapeia as declarações OIDC para as chaves de contexto de condição do AWS STS na AWS. É possível usar essas chaves para controlar o acesso a um perfil. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

AWS STSChave da condição	declaração JWT do IdP	Disponível na sessão
organization_slug	organization_slug	Não
organization_id	organization_id	Não
pipeline_slug	pipeline_slug	Não
pipeline_id	pipeline_id	Não
cluster_name	cluster_name	Não
cluster_id	cluster_id	Não
build_branch	build_branch	Não

organization_slug

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:organization_slug`

Essa chave identifica a organização do Buildkite na qual o trabalho é executado. Use-a para restringir o acesso a organizações específicas.

organization_id

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:organization_id`

Essa chave verifica o ID da organização do Buildkite na qual o trabalho é executado. Os IDs da organização são gerados pelo Buildkite e são imutáveis.

pipeline_slug

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:pipeline_slug`

Essa chave identifica o pipeline no qual o trabalho é executado. Use-a para restringir o acesso a pipelines do Buildkite específicos.

pipeline_id

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:pipeline_id`

Essa chave verifica o ID do pipeline no qual o trabalho está sendo executado. Os IDs de pipeline são gerados pelo Buildkite e são imutáveis.

cluster_name

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:cluster_name`

Essa chave identifica o cluster no qual o trabalho é executado. Use-a para restringir o acesso a clusters do Buildkite específicos.

cluster_id

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:cluster_id`

Essa chave verifica o ID do cluster no qual o trabalho está sendo executado. Os IDs de cluster são gerados pelo Buildkite e são imutáveis.

build_branch

Funciona com operadores de string.

Exemplo – `agent.buildkite.com:build_branch`

Essa chave identifica a ramificação do git que acionou a compilação. Use-a para limitar o acesso com base em ramificações específicas, como permitir somente as ramificações `main` ou `production`.

> **Use identificadores imutáveis, não slugs**
>
> No Buildkite, uma organização ou slug de pipeline excluída pode ser imediatamente registrada novamente por uma organização ou pipeline diferente. O Buildkite emite tokens OIDC para esses slugs do mesmo emissor (`agent.buildkite.com`). Como resultado, as políticas que dependem exclusivamente de reivindicações baseadas em slugs (como `organization_slug` ou `pipeline_slug`) podem conceder acesso a identidades não intencionais. Para ajudar a se proteger contra isso, use identificadores estáveis e imutáveis, como `organization_id` ou `pipeline_id`, em suas políticas de confiança de perfil ou políticas de controle de recursos.

O exemplo de política de confiança a seguir usa declarações personalizadas no token OIDC do Buildkite para limitar o acesso a um perfil.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "Federated": "arn:aws:iam::123456789012:oidc-provider/
agent.buildkite.com"
        },
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {
```

```
                "agent.buildkite.com:aud": "sts.amazonaws.com",
                "agent.buildkite.com:organization_id": "0191e7a0-1234-7abc-
def0-123456789abc",
                "agent.buildkite.com:pipeline_id": "0191e7a1-5678-7abc-
def0-123456789abc",
                "agent.buildkite.com:build_branch": "main"
            }
        }
    }
    ]
}
```

Mais informações sobre a federação OIDC

- Guia do usuário do Amazon Cognito
- Federação OIDC

Chaves disponíveis para federação do AWS STS com base em SAML

Se você estiver trabalhando com federação baseada em SAML usando o AWS Security Token Service (AWS STS), poderá incluir chaves de condição adicionais na política.

Políticas de confiança da função do SAML

Na política de confiança de uma função, você pode incluir as seguintes chaves, que ajudam você a estabelecer se o chamador tem permissão para assumir a função. Exceto no caso de `saml:doc`, todos os valores são derivados da declaração de SAML. Todos os itens da lista estão disponíveis no editor visual do console do IAM quando você cria ou edita uma política com condições. Os itens marcados com `[]` podem ter um valor que seja uma lista do tipo especificado.

saml:aud

Funciona com operadores de string.

Um URL de endpoint para a qual as declarações de SAML são apresentadas. O valor dessa chave vem do campo `SAML Recipient` na declaração, nãodo campo `Audience`.

saml:commonName[]

Funciona com operadores de string.

Esse é um atributo `commonName`.

saml:cn[]

Funciona com operadores de string.

Esse é um atributo `eduOrg`.

saml:doc

Funciona com operadores de string.

Isso representa a entidade principal que foi usada para assumir a função. O formato é *`account-ID`*/*`provider-friendly-name`*, como `123456789012/SAMLProviderName`. O valor account-ID se refere à conta que possui o provedor SAML.

saml:edupersonaffiliation[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonassurance[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonentitlement[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonnickname[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonorgdn

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonorgunitdn[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonprimaryaffiliation

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonprimaryorgunitdn

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonprincipalname

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersonscopedaffiliation[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:edupersontargetedid[]

Funciona com operadores de string.

Esse é um atributo `eduPerson`.

saml:eduorghomepageuri[]

Funciona com operadores de string.

Esse é um atributo `eduOrg`.

saml:eduorgidentityauthnpolicyuri[]

Funciona com operadores de string.

Esse é um atributo `eduOrg`.

saml:eduorglegalname[]

Funciona com operadores de string.

Esse é um atributo `eduOrg`.

saml:eduorgsuperioruri[]

Funciona com operadores de string.

Esse é um atributo `eduOrg`.

saml:eduorgwhitepagesuri[]

Funciona com operadores de string.

Esse é um atributo `eduOrg`.

saml:givenName[]

Funciona com operadores de string.

Esse é um atributo `givenName`.

saml:iss

Funciona com operadores de string.

O emissor, que é representado por um URN.

saml:mail[]

Funciona com operadores de string.

Esse é um atributo `mail`.

saml:name[]

Funciona com operadores de string.

Esse é um atributo `name`.

saml:namequalifier

Funciona com operadores de string.

Um valor de hash baseado no nome amigável do provedor SAML. O valor é a concatenação dos seguintes valores, em ordem e separados por um caractere '/':

1. O valor da resposta `Issuer` (`saml:iss`)
2. O ID da conta da `AWS`.

3. O nome amigável (a última parte do ARN) do provedor SAML no IAM

A concatenação do ID da conta e do nome amigável do provedor SAML está disponível para as políticas do IAM como a chave `saml:doc`. Para obter mais informações, consulte Identificar exclusivamente os usuários na federação baseada em SAML.

saml:organizationStatus[]

Funciona com operadores de string.

Esse é um atributo `organizationStatus`.

saml:primaryGroupSID[]

Funciona com operadores de string.

Esse é um atributo `primaryGroupSID`.

saml:sub

Funciona com operadores de string.

Trata-se do assunto da solicitação, que inclui um valor que identifica de forma exclusiva um usuário individual em uma organização (por exemplo, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Funciona com operadores de string.

Essa chave pode ter o valor `persistent`, `transient` ou consistir no URI `Format` completo dos elementos `Subject` e `NameID` usados em sua declaração SAML. O valor `persistent` indica que o valor em `saml:sub` é o mesmo para um usuário entre as sessões. Se o valor for `transient`, o usuário terá um valor `saml:sub` diferente para cada sessão. Para obter mais informações sobre o atributo `NameID` do elemento `Format`, consulte Configurar declarações SAML para a resposta de autenticação.

saml:surname[]

Funciona com operadores de string.

Esse é um atributo `surnameuid`.

saml:uid[]

Funciona com operadores de string.

Esse é um atributo `uid`.

saml:x500UniqueIdentifier[]

Funciona com operadores de string.

Esse é um atributo `x500UniqueIdentifier`.

Para obter informações gerais sobre os atributos do `eduPerson` e `eduOrg`, consulte o REFEDS Wiki website (site Wiki do REFEDS). Para obter uma lista de atributos do `eduPerson`, consulte a eduPerson Object Class Specification (201602) (Especificação da classe de objeto eduPerson (201602)).

As chaves de condição cujo tipo é uma lista pode incluir vários valores. Para criar condições na política para listar os valores, você pode usar operadores definidos (`ForAllValues`, `ForAnyValue`). Por exemplo, para permitir qualquer usuário cuja afiliação seja "corpo docente" ou "equipe" (mas não "aluno"), você pode usar uma condição como a seguinte:

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Chaves de contexto de federação do AWS STS baseadas em SAML entre serviços

Algumas chaves de condição de federação baseadas em SAML podem ser usadas em solicitações subsequentes para autorizar operações da AWS em outros serviços e chamadas `AssumeRole`. Essas são as chaves de condição que podem ser usadas em políticas de confiança de perfil quando as entidades principais federadas assumem outro perfil, e em políticas de recursos de outros serviços da AWS para autorizar o acesso das entidades principais federadas aos recursos. Para obter mais informações sobre o uso dessas chaves, consulte Sobre federação baseada em SAML 2.0.

Selecione uma chave de condição para ver a descrição.

- saml:namequalifier
- saml:sub
- saml:sub_type

> **Note**
>
> Nenhuma outra chave de condição de federação baseada em SAML está disponível para uso após a resposta inicial de autenticação do provedor de identidades (IdP) externo.

Chaves disponíveis do AWS STS

É possível usar as chaves de condição a seguir nas políticas de confiança de função do IAM para funções que são assumidas usando operações do AWS Security Token Service (AWS STS).

saml:sub

Funciona com operadores de string.

Trata-se do assunto da solicitação, que inclui um valor que identifica de forma exclusiva um usuário individual em uma organização (por exemplo, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

sts:AWSServiceName

Funciona com operadores de string.

Use essa chave para especificar um serviço onde um token de portador pode ser usado. Ao usar essa chave de condição em uma política, especifique o serviço usando um principal de serviço. Um serviço principal é o nome de um serviço que pode ser especificado no elemento `Principal` de uma política. Por exemplo, `codeartifact.amazonaws.com` é a entidade principal do AWS CodeArtifact.

Disponibilidade: essa chave está presente em solicitações que recebem um token de portador. Não é possível fazer uma chamada direta ao AWS STS para obter um token de portador. Quando você executa algumas operações em outros serviços, o serviço solicita o token de portador em seu nome.

Alguns serviços da AWS exigem que você tenha permissão para obter um token de portador do serviço AWS STS para poder acessar seus recursos de forma programática. Por exemplo, o AWS CodeArtifact requer que as entidades principais usem tokens de portador para realizar algumas operações. O comando `aws codeartifact get-authorization-token` retorna um token de portador. É possível usar o token do portador para realizar as operações do AWS CodeArtifact. Para obter mais informações sobre tokens de portador, consulte Tokens de portador de serviço.

É possível usar essa chave de condição para permitir que os principais obtenham um token de portador para uso com um serviço específico.

sts:DurationSeconds

Funciona com operadores numéricos.

Use esta chave para especificar a duração (em segundos) que uma entidade principal pode usar ao obter um token de portador do AWS AWS STS ou um JSON Web Token da API GetWebIdentityToken.

Disponibilidade: esta chave está presente em solicitações que recebem um token de portador ou um JSON Web Token da API GetWebIdentityToken. Não é possível fazer uma chamada direta ao AWS STS para obter um token de portador. Quando você executa algumas operações em outros serviços, o serviço solicita o token de portador em seu nome. A chave não é aplicável para operações de perfil assumido do AWS STS.

Alguns serviços da AWS exigem que você tenha permissão para obter um token de portador do serviço AWS STS para poder acessar seus recursos de forma programática. Por exemplo, o AWS CodeArtifact requer que as entidades principais usem tokens de portador para realizar algumas operações. O comando `aws codeartifact get-authorization-token` retorna um token de portador. É possível usar o token do portador para realizar as operações do AWS CodeArtifact. Para obter mais informações sobre tokens de portador, consulte Tokens de portador de serviço.

sts:IdentityTokenAudience

Funciona com operadores de string.

Use esta chave para especificar o público para o qual uma entidade principal do IAM pode solicitar JSON Web Tokens (JWTs) usando a API GetWebIdentityToken. Quando essa chave de condição está presente em uma política do IAM, as entidades principais do IAM só podem solicitar tokens para os públicos especificados na política. Os serviços externos validam a declaração audience ("aud") no JSON Web Token para certificar-se de que o token foi destinado a eles.

Disponibilidade: esta chave está presente nas solicitações à API GetWebIdentityToken, que é usada para obter JSON Web Tokens (JWTs) para autenticação com serviços externos.

Ao usar esta chave de condição em uma política, especifique o valor de audience que corresponda ao identificador do destinatário pretendido (por exemplo, https://api.example.com).

A seguinte política de exemplo permite que uma entidade principal solicite tokens para os serviços externos especificados:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "sts:IdentityTokenAudience": [
                        "https://api2.example.com",
                        "https://api1.example.com"
                    ]
                }
            }
        }
    ]
}
```

sts:SigningAlgorithm

Funciona com operadores de string.

Use esta chave para especificar o algoritmo criptográfico que o AWS AWS STS usa para assinar JSON Web Tokens (JWTs) gerados pela API GetWebIdentityToken. Ao usar essa chave de condição em uma política, especifique ES384 (ECDSA com curva P-384 e SHA-384) ou RS256 (RSA com SHA-256).

Disponibilidade: esta chave está presente nas solicitações à API GetWebIdentityToken, que é usada para obter JSON Web Tokens (JWTs) para autenticação com serviços externos.

Você pode usar essa chave de condição para garantir que as entidades principais do IAM solicitem tokens usando algoritmos de assinatura compatíveis com seus requisitos de segurança ou com os serviços externos com os quais você se integra. O ES384 oferece segurança e desempenho ideais, enquanto o RS256 oferece compatibilidade mais ampla com sistemas que não compatíveis com o ECDSA.

O exemplo de política a seguir exige que as entidades principais usem o algoritmo de assinatura ES384:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:GetWebIdentityToken",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "sts:SigningAlgorithm": "ES384"
                }
            }
        }
    ]
}
```

sts:ExternalId

Funciona com operadores de string.

Use essa chave para exigir que uma entidade principal forneça um identificador específico ao assumir um perfil do IAM.

Disponibilidade: essa chave está presente na solicitação quando a entidade principal fornece um ID externo enquanto assume um perfil usando a AWS CLI ou a API da AWS.

Um identificador exclusivo que pode ser necessário ao assumir uma função em outra conta. Se o administrador da conta à qual a função pertence forneceu um ID externo para você, forneça esse valor no parâmetro `ExternalId`. Esse valor pode ser qualquer string, como uma frase secreta ou o número de uma conta. A função principal do ID externo é abordar e impedir o problema "confused deputy". Para obter mais informações sobre o ID externo e o problema confused deputy, consulte Acesso às Contas da AWS de propriedade de terceiros.

O valor `ExternalId` deve ter no mínimo 2 e no máximo 1.224 caracteres. O valor deve ser alfanumérico sem espaço em branco. Ele também pode incluir os seguintes símbolos: mais (+), igual (=), vírgula (,), ponto (.), arroba (@), dois pontos (:), barra (/) e hífen (-).

sts:RequestContext/chave de contexto

Funciona com operadores de string.

Use essa chave para comparar os pares de chave-valor do contexto da sessão que estão incorporados na declaração de contexto assinada pelo emissor do token de confiança transmitida

na solicitação com os valores de chave de contexto especificados na política de confiança do perfil.

Disponibilidade: essa chave está presente na solicitação quando uma declaração de contexto é fornecida no parâmetro `ProvidedContexts` da solicitação enquanto assume um perfil usando a operação de API AssumeRole do AWS STS.

Essa chave de contexto é formatada como `"sts:RequestContext/context-key":"context-value"`, onde `context-key` e `context-value` são um par de chave/valor de contexto. Quando várias chaves de contexto são incorporadas na declaração de contexto assinada que é passada na solicitação, existe uma chave de contexto para cada par de chave/valor. Você deve conceder permissão para a ação `sts:SetContext` na política de confiança do perfil para permitir que uma entidade principal defina chaves de contexto no token de sessão resultante. Para saber mais sobre as chaves de contexto do Centro de Identidade do IAM compatíveis que podem ser usadas com essa chave, consulte Chaves de condição do AWS STS para o Centro de Identidade do IAM no Guia do usuário do Centro de Identidade do AWS IAM.

É possível usar essa chave em uma política de confiança do perfil para aplicar um controle de acesso refinado com base no usuário ou em seus atributos ao assumir um perfil. Depois que o perfil é assumido, a atividade aparece nos logs do AWS CloudTrail no atributo `AdditionalEventData`, contendo os pares de chave/valor de contexto da sessão que foram definidos pelo provedor de contexto na solicitação feita para assumir o perfil. Isso torna mais fácil para os administradores diferenciar entre sessões de função quando uma função é usada por diferentes entidades de segurança. Os pares de chave/valor são definidos pelo provedor de contexto especificado, não pelo AWS CloudTrail ou pelo AWS STS. Isso dá ao provedor de contexto controle sobre qual contexto é incluído nos logs e nas informações da sessão do CloudTrail.

sts:RequestContextProviders

Funciona com operadores de nome de recurso da Amazon (ARN).

Use essa chave para comparar o ARN do provedor de contexto na solicitação com o ARN do provedor de contexto especificado na política de confiança do perfil.

Disponibilidade: essa chave está presente na solicitação quando uma declaração de contexto é fornecida no parâmetro `ProvidedContexts` da solicitação enquanto assume um perfil usando a operação de API AssumeRole do AWS STS.

O exemplo de condição a seguir verifica se o ARN do provedor de contexto passado na solicitação corresponde ao ARN especificado na condição da política de confiança do perfil. Recomendamos que você adicione uma verificação nula com `ForAllValues` para evitar que chaves de contexto ausentes ou chaves de contexto com valores vazios sejam avaliadas como verdadeiras. Para obter detalhes, consulte Operador de condição para verificar a existência de chaves de condição.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Action": "sts:SetContext",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
      "ForAllValues:ArnEquals": {
        "sts:RequestContextProviders": [
          "arn:aws:iam::aws:contextProvider/IdentityCenter"
        ]
      },
      "Null": {
        "sts:RequestContextProviders": "false"
      }
    }
  }
}
```

sts:RoleAuthorizedByIdp

Funciona com Operadores boolianos.

Use essa chave para verificar se o provedor de identidades (IdP) autorizou explicitamente o perfil solicitado por meio da declaração `https://aws.amazon.com/roles` no token OIDC. A declaração de perfis contém os ARNs de perfil do IAM que o token pode assumir. O AWS STS suporta dois formatos para o valor da declaração:

Matriz JSON (recomendada):

```
"https://aws.amazon.com/roles": [
```

```
  "arn:aws:iam::111122223333:role/RoleA",
  "arn:aws:iam::111122223333:role/RoleB"
]
```

String delimitada por ponto e vírgula:

```
"https://aws.amazon.com/roles": "arn:aws:iam::111122223333:role/
RoleA;arn:aws:iam::111122223333:role/RoleB"
```

Disponibilidade — Essa chave está presente nas solicitações que usam a operação `AssumeRoleWithWebIdentity`. O valor é `true` quando o token inclui a declaração `https://aws.amazon.com/roles` e o ARN do perfil solicitado corresponde a um valor na declaração. O valor é `false` quando o token não inclui a declaração. Como a chave está sempre presente nas solicitações `AssumeRoleWithWebIdentity`; use o operador de condição `Bool` em vez de `BoolIfExists`.

Quando o token inclui a declaração, mas o ARN do perfil solicitado não corresponde a um valor na declaração, a chamada `AssumeRoleWithWebIdentity` retorna um erro `InvalidIdentityToken`.

Você pode usar essa chave de condição em políticas de confiança de perfil, políticas de controle de recursos (RCPs) e políticas de endpoint da VPC para exigir que um IdP autorize o perfil antes que ele possa ser assumido.

A seguinte política de confiança de perfil permite `AssumeRoleWithWebIdentity` somente quando o IdP autorizar o perfil solicitado no token.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
idp.example.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": { "idp.example.com:aud": "my-app-id" },
                "Bool": { "sts:RoleAuthorizedByIdp": "true" }
            }
```

```
        }
    ]
}
```

sts:RoleSessionName

Funciona com operadores de string.

Utilize essa chave para comparar o nome da sessão que um principal especifica ao assumir uma função com o valor especificado na política.

Disponibilidade: essa chave está presente na solicitação quando a entidade principal assume o perfil usando o Console de gerenciamento da AWS, qualquer comando de assumir função da CLI ou qualquer operação de API `AssumeRole` do AWS STS.

É possível usar essa chave em uma política de confiança de perfil para exigir que os usuários forneçam um nome de sessão específico quando assumirem um perfil. Por exemplo, você pode exigir que os usuários do IAM especifiquem o próprio nome de usuário como nome de sessão. Depois que o usuário do IAM assume a função, a atividade aparece nos logs do AWS CloudTrail com o nome da sessão que corresponde ao seu nome de usuário. Isso torna mais fácil para os administradores diferenciar entre sessões de função quando uma função é usada por diferentes entidades de segurança.

A política de confiança de função a seguir exige que os usuários do IAM na conta `111122223333` forneçam o nome de usuário do IAM como o nome da sessão quando assumem a função. Esse requisito é imposto usando a variável de condição `aws:username` na chave de condição. Essa política permite que os usuários do IAM assumam a função à qual a política está anexada. Essa política não permite que ninguém que utilize credenciais temporárias assuma a função, porque a variável `username` está presente apenas para usuários do IAM.

> **Important**
>
> É possível usar qualquer chave de condição de valor único disponível como uma variável. Você não pode usar uma chave de condição de valores múltiplos como uma variável.

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "prefix-
${aws:username}"}
            }
        }
    ]
}
```

Quando um administrador exibe o log do AWS CloudTrail de uma ação, ele pode comparar o nome da sessão com os nomes de usuário em sua conta. No exemplo a seguir, o usuário chamado `matjac` executou a operação usando a função chamada `MateoRole`. O administrador pode entrar em contato com Mateo Jackson, que tem o nome de usuário `matjac`.

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

Se você permitir acesso entre contas usando funções, os usuários em uma conta poderão assumir uma função em outra conta. O ARN do usuário da função assumida listado no CloudTrail inclui a conta onde a função existe. Ele não inclui a conta do usuário que assumiu a função. Os usuários são exclusivos apenas dentro de uma conta. Portanto, recomendamos que você use esse método para verificar logs do CloudTrail somente para funções que são assumidas pelos usuários em contas que você administra. Seus usuários podem usar o mesmo nome de usuário em várias contas.

sts:SourceIdentity

Funciona com operadores de string.

Use esta chave para comparar a identidade-fonte que uma entidade principal especifica ao assumir um perfil com o valor que é especificado na política.

Disponibilidade: esta chave está presente na solicitação quando a entidade principal fornece uma identidade-fonte enquanto assume um perfil usando qualquer comando de assumir perfil da CLI do AWS STS ou operação da API `AssumeRole` do AWS STS.

É possível usar essa chave em uma política de confiança de perfil para exigir que seus usuários definam uma identidade-fonte específica ao assumir um perfil. Por exemplo, você pode exigir que seu quadro de funcionários ou suas identidades federadas especifiquem um valor para a identidade-fonte. É possível configurar seu provedor de identidade (IdP) para usar um dos atributos associados aos usuários, como um nome de usuário ou e-mail como a identidade-fonte. O IdP então passa a identidade-fonte como um atributo nas declarações ou reivindicações que envia para a AWS. O valor do atributo de identidade-fonte identifica o usuário ou a aplicação que está assumindo a função.

Depois que o usuário assume a função, a atividade aparece nos logs do AWS CloudTrail com o valor de identidade-fonte que foi definido. Isso torna mais fácil para os administradores determinar quem ou o que executou ações com uma função na AWS. Você deve conceder permissões para a ação `sts:SetSourceIdentity` para permitir que uma identidade defina uma identidade-fonte.

Ao contrário de `sts:RoleSessionName`, após a definição da identidade-fonte, o valor não pode ser alterado. Ele está presente no contexto de solicitação para todas as ações executadas com a função pela identidade-fonte. O valor persiste nas sessões de função subsequentes quando você usa as credenciais da sessão para assumir outra função. Assumir uma função de outra é chamado de encadeamento de funções.

É possível usar a chave de condição global `aws:SourceIdentity` para controlar ainda mais o acesso a recursos da AWS com base no valor da identidade-fonte em solicitações subsequentes.

A política de confiança de função a seguir permite que o usuário do IAM `AdminUser` assuma uma função na conta `111122223333`. Ela também concede permissão para o `AdminUser` definir uma identidade-fonte, desde que o conjunto de identidades de origem seja `DiegoRamirez`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
```

```
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

Para saber mais sobre como usar informações de identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

sts:TaskPolicyArn

Funciona com operadores de nome de recurso da Amazon (ARN).

Use essa chave para comparar o ARN da política em uma solicitação sts:AssumeRoot com o ARN da política especificado na política.

Disponibilidade: essa chave está presente na solicitação quando você faz uma solicitação usando sts:AssumeRoot.

Os administradores podem usar essa chave de condição nas políticas do IAM para restringir funções ou usuários específicos da conta gerencial ou da conta de administrador delegado de realizar determinadas ações ao assumir as credenciais raiz. Para obter mais informações, consulte Execução de uma tarefa privilegiada em uma conta-membro do AWS Organizations.

sts:TransitiveTagKeys

Funciona com operadores de string.

Use esta chave para comparar as chaves de tag de sessão transitiva na solicitação com as especificadas na política.

Disponibilidade: essa chave está presente na solicitação quando você faz uma solicitação usando credenciais de segurança temporárias. Elas incluem credenciais criadas usando qualquer operação assume-role, ou a operação `GetFederationToken`.

Ao fazer uma solicitação usando credenciais de segurança temporárias, o contexto da solicitação inclui a chave de contexto `aws:PrincipalTag`. Essa chave inclui uma lista de tags de sessão, tags de sessão transitivas e tags de função. As tags de sessão transitivas são tags que persistem em todas as sessões subsequentes quando você usa as credenciais da sessão para assumir outra função. Assumir uma função de outra é chamado de encadeamento de funções.

É possível usar essa chave de condição em uma política para exigir a configuração de tags de sessão específicas como transitivas ao assumir um perfil ou federar um usuário.

Ações, recursos e chaves de condição dos serviços da AWS

Cada serviço da AWS pode definir ações, recursos e chaves de contexto de condição para uso em políticas do IAM. Para obter uma lista de serviços do AWS e suas ações, recursos e chaves de contexto de condição, consulte Ações, recursos e chaves de condição na documentação de Referência de autorização de serviço.

Recursos para saber mais sobre o IAM

O IAM é um produto avançado, nele, você encontrará muitos recursos que ajudarão a aprender mais sobre como esse serviço auxiliará na proteção da sua Conta da AWS e seus recursos.

Tópicos

- Identidades
- Credenciais (senhas, chaves de acesso e dispositivos com MFA)
- Políticas e permissões
- Federação e delegação
- IAM e outros produtos da AWS
- Práticas de segurança gerais
- Recursos gerais da

Identidades

Consulte estes recursos para criar, gerenciar e usar identidades.

- Manage identities in IAM Identity Center (Gerenciar identidades no Centro de Identidade do IAM): informações processuais sobre a criação de usuários e grupos no Centro de Identidade do IAM.
- Identidades do IAM: uma discussão aprofundada sobre usuários, grupos e perfis.

Credenciais (senhas, chaves de acesso e dispositivos com MFA)

Analise os guias a seguir para gerenciar senhas, chaves de acesso e dispositivos de MFA para sua Conta da AWS e usuários do IAM.

- Senhas de usuários na AWS: descreve as opções de gerenciamento de senhas para usuários do IAM em sua conta.
- Gerenciar chaves de acesso para usuários do IAM: descreve como as chaves de acesso funcionam e como usá-las para fazer chamadas programáticas para a AWS. Existem alternativas mais seguras às chaves de acesso que recomendamos considerar primeiro. Para obter mais informações, consulte Considerações e alternativas para chaves de acesso de longo prazo no Guia de Referência geral da AWS.

- AWSCódigo da autenticação multifator no IAM da – descreve como configurar sua conta e os usuários do IAM para exigir uma senha e um código de uso único gerado em um dispositivo antes que o login seja permitido. (Isso às vezes é chamado de autenticação de dois fatores.)

Para obter informações gerais sobre os tipos de credenciais que você usa para acessar a Amazon Web Services, consulte Credenciais de segurança da AWS, no Guia de Referência geral da AWS.

Políticas e permissões

Saiba o funcionamento interno das políticas do IAM e encontre dicas sobre as melhores maneiras de conceder permissões:

- Políticas e permissões no AWS Identity and Access Management – apresenta a linguagem de política que é usada para definir permissões. Descreve como as permissões podem ser anexadas a usuários ou grupos ou, para alguns produtos da AWS, aos próprios recursos.
- Referência de elemento de política JSON do IAM – fornece descrições e exemplos de cada elemento da linguagem de política.
- Validação de política do IAM – localiza recursos para validação da política JSON.
- Exemplos de políticas baseadas em identidade do IAM: mostra exemplos de políticas para tarefas comuns em vários produtos da AWS.
- AWS Policy Generator: crie políticas personalizadas escolhendo produtos e ações de uma lista.
- Simulador de políticas do IAM: teste se uma política permitiria ou negaria uma solicitação específica para a AWS.

Federação e delegação

Você pode conceder acesso a recursos na sua Conta da AWS para usuários que foram autenticados (fizeram login) em outro lugar. Eles podem ser usuários do IAM em outra Conta da AWS (conhecido como delegação), usuários que foram autenticados com o processo de login da sua organização ou usuários de um provedor de identidade da Internet, como Login with Amazon, Facebook, Google ou qualquer outro provedor de identidade compatível com OpenID Connect (OIDC). Nesses casos, os usuários obtêm credenciais de segurança temporárias para acessar recursos da AWS.

- Tutorial do IAM: Delegar acesso entre contas da AWS usando funções do IAM – orienta você pelo processo de concessão de acesso entre contas a um usuário do IAM em outra Conta da AWS.

- Cenários comuns para credenciais temporárias: descreve as maneiras pelas quais os usuários podem ser federados na AWS após serem autenticados fora da AWS.

IAM e outros produtos da AWS

A maioria dos produtos da AWS é integrada ao IAM, de modo que você pode usar recursos do IAM para ajudar a proteger o acesso aos recursos nesses produtos. Os recursos a seguir abordam o IAM e a segurança de alguns dos produtos mais populares produtos da AWS. Para obter uma lista completa de produtos que funcionam com o IAM, incluindo links para obter mais informações sobre cada um, consulte AWSServiços da que funcionam com o IAM.

Uso do IAM com o Amazon EC2

- Controle do acesso aos recursos do Amazon EC2: descreve como usar recursos do IAM para permitir que os usuários administrem instâncias, volumes e muito mais do Amazon EC2.
- Usar perfis de instância: descreve como usar as funções do IAM para fornecer credenciais de forma segura a aplicações que são executadas em instâncias do Amazon EC2 e que precisam de acesso a outros produtos da AWS.

Uso do IAM com o Amazon S3

- Gerenciamento de permissões de acesso para seus recursos do Amazon S3: discute o modelo de segurança do Amazon S3 para buckets e objetos, o que inclui políticas do IAM.
- Escrever políticas do IAM: conceder acesso a pastas específicas do usuário em um bucket do Amazon S3: discute como permitir que os usuários protejam suas próprias pastas no Amazon S3. (Para obter mais publicações sobre o Amazon S3 e o IAM, escolha a etiqueta S3 abaixo do título da publicação do blog.)

Uso do IAM com o Amazon RDS

- Uso do AWS Identity and Access Management (IAM) para gerenciar o acesso aos recursos do Amazon RDS: descreve como usar o IAM para controlar o acesso a instâncias de banco de dados, snapshots de banco de dados e muito mais.
- Um primer nas permissões no nível do recurso do RDS: descreve como usar o IAM para controlar o acesso a instâncias específicas do Amazon RDS.

Uso do IAM com o Amazon DynamoDB

- Uso do IAM para controlar o acesso aos recursos do DynamoDB: descreve como usar o IAM para permitir que os usuários administrem tabelas e índices do DynamoDB.
- O vídeo a seguir (8:55) explica como fornecer controle de acesso para itens ou atributos individuais (ou ambos) do banco de dados DynamoDB.

  Getting Started with Fine-Grained Access Control for DynamoDB

Práticas de segurança gerais

Encontre dicas e orientações de especialistas sobre as melhores maneiras de proteger recursos e sua Conta da AWS:

- Práticas recomendadas de segurança, identidade e conformidade: encontre recursos para gerenciar a segurança entre Contas da AWS e produtos, incluindo sugestões de arquitetura de segurança, uso do IAM, criptografia e segurança de dados e muito mais.
- Identity and Access Management: o AWS Well-Architected Framework ajuda você a entender os principais conceitos, princípios de design e práticas recomendadas de arquitetura para projetar e executar workloads na nuvem.
- Práticas recomendadas de segurança no IAM: oferece recomendações sobre maneiras de usar o IAM para ajudar a proteger sua Conta da AWS e recursos.
- Guia do usuário do AWS CloudTrail: use o AWS CloudTrail para rastrear um histórico de chamadas de API feitas para a AWS e armazenar essas informações em arquivos de log. Isso ajuda você a determinar quais usuários e contas acessaram recursos na sua conta, quando as chamadas foram feitas, quais ações foram solicitadas, etc.

Recursos gerais da

Explore os recursos a seguir para saber mais sobre o IAM e a AWS.

- Informações do produto IAM: informações gerais sobre o produto AWS Identity and Access Management.
- AWS re:Post para AWS Identity and Access Management: visite AWS re:Post para discutir questões técnicas relacionadas ao IAM com a comunidade da AWS.

- Aulas e workshops — Links para cursos de especialidades e baseados em perfil, bem como laboratórios autoguiados para ajudar a aperfeiçoar suas habilidades na AWS e a obter experiência prática.
- Centro dos desenvolvedores da AWS — Explore tutoriais, baixe ferramentas e informe-se sobre eventos para desenvolvedores da AWS.
- Ferramentas do desenvolvedor da AWS — Links para ferramentas de desenvolvedor, SDKs, toolkits de IDE e ferramentas da linha de comando para desenvolver e gerenciar aplicações da AWS.
- Centro de recursos de conceitos básicos — Saiba como configurar a Conta da AWS, participar da comunidade da AWS e lançar sua primeira aplicação.
- Tutoriais práticos — Siga os tutoriais passo a passo para iniciar sua primeira aplicação na AWS.
- Whitepapers da AWS — Links para uma lista abrangente de whitepapers técnicos da AWS que abrangem tópicos como arquitetura, segurança e economia, elaborados pelos arquitetos de soluções da AWS ou por outros especialistas técnicos.
- AWS Support Center – a central para criar e gerenciar seus casos do AWS Support. Também inclui links para outros recursos úteis, como fóruns, perguntas frequentes técnicas, status de integridade do serviço e AWS Trusted Advisor.
- Suporte: a página Web principal para obter informações sobre o Suporte, um canal de suporte de resposta rápida e com atendimento individual para ajudar a construir e a executar aplicações na nuvem.
- Entrar em contato: um ponto central de contato para consultas relativas a faturas da AWS, contas, eventos, uso abusivo e outros problemas.
- Termos do site da AWS – informações detalhadas sobre nossos direitos autorais e marca registrada; sua conta, licença e acesso ao site, entre outros tópicos.

Chamar a API do IAM usando solicitações de consulta HTTP

Conteúdo

- Endpoints
- HTTPS obrigatório
- Assinar solicitações de API do IAM

É possível acessar os serviços IAM e AWS STS de forma programática usando a API de consulta. As solicitações da API de consulta são solicitações HTTPS que devem conter um parâmetro `Action` para indicar a ação a ser realizada. O IAM e o AWS STS dão suporte a solicitações GET e POST para todas as ações. Ou seja, a API não exige que você use GET para algumas ações e POST para outras. No entanto, solicitações GET estão sujeitas à limitação de tamanho de um URL, embora esse limite dependa do navegador, um limite típico é de 2.048 bytes. Portanto, para as solicitações da API de consulta que exigem tamanhos maiores, você deve usar uma solicitação POST.

A resposta é um documento XML. Para obter detalhes sobre a resposta, consulte as páginas de ação individual na Referência da API do IAM ou na Referência da API do AWS Security Token Service.

> **Tip**
>
> Em vez de fazer chamadas diretas para as operações da API do IAM ou do AWS STS, você pode usar um dos AWS SDKs. Os AWS SDKs consistem em bibliotecas e código de exemplo de várias linguagens de programação e plataformas (Java, Ruby, .NET, iOS, Android etc.). Os SDKs constituem uma forma conveniente de criar acesso programático para o IAM e a AWS. Por exemplo, os SDKs processam tarefas como assinatura criptográfica de solicitações (veja abaixo), gerenciamento de erros e novas tentativas automáticas de solicitações. Para obter informações sobre os AWS SDKs, incluindo como fazer download deles e instalá-los, consulte a página Ferramentas para a Amazon Web Services.

Para obter mais detalhes sobre as ações da API e os erros, consulte a Referência da API do IAM ou a Referência da API do AWS Security Token Service.

Endpoints

O IAM e o AWS STS têm, cada um, um único endpoint global:

- (IAM) https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Important**
>
> O AWS STS também é compatível com o envio de solicitações aos endpoints regionais, além de ao endpoint global. A AWS recomenda o uso de endpoints regionais em vez de endpoints globais para reduzir a latência, incorporar redundância e aumentar a validade dos tokens de sessão. Para poder usar o AWS STS em uma região, você deve primeiro ativar o STS nessa região para a sua Conta da AWS. Para obter mais informações sobre como ativar regiões adicionais para o AWS STS, consulte Gerenciar o AWS STS em uma Região da AWS.

Para obter mais informações sobre endpoints e regiões da AWS para todos os serviços, consulte Cotas e endpoints de serviço na Referência geral da AWS.

HTTPS obrigatório

Como a API de consulta retorna informações confidenciais, como credenciais de segurança, você deve usar HTTPS com todas as solicitações de API.

Assinar solicitações de API do IAM

As solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta. É altamente recomendado que você não use as credenciais da Usuário raiz da conta da AWS para tarefas diárias com o IAM. Você pode usar as credenciais para um usuário do IAM ou usar o AWS STS para gerar credenciais de segurança temporárias.

Para assinar suas solicitações de API, recomendamos o uso do AWS Signature Version 4. Para obter informações sobre como usar o Signature versão 4, consulte Processo de assinatura do Signature versão 4 na Referência geral da AWS.

Se você precisar usar o Signature Version 2, informações sobre como usá-lo estão disponíveis na Referência geral da AWS.

Para saber mais, consulte:

- AWS Credenciais de segurança da. Fornece informações gerais sobre os tipos de credenciais usadas para acessar a AWS.
- Práticas recomendadas de segurança no IAM. Apresenta uma lista de sugestões para usar o serviço IAM para ajudar a proteger seus recursos da AWS.
- Credenciais de segurança temporárias no IAM. Descreve como criar e usar credenciais de segurança temporárias.

Histórico de documentos do IAM

A tabela a seguir descreve as principais atualizações da documentação do IAM.

Alteração	Descrição	Data
job_workflow_ref aceita como reivindicação de definição de escopo alternativa para o provedor OIDC do GitHub Actions	Em relação ao provedor OIDC do GitHub Actions, o IAM aceita a partir de agora a reivindicação `sub` ou a reivindicação `job_workflow_ref` como controle do provedor de identidade nas políticas de confiança de perfil.	12 de maio de 2026
Chaves de contexto de condição do IAM para APIs de credenciais específicas do serviço	O IAM agora oferece suporte a duas novas chaves de contexto de condição para controlar o acesso às APIs de credenciais específicas do serviço: `iam:ServiceSpecificCredentialAgeDays` e `iam:ServiceSpecificCredentialServiceName`. Essas chaves permitem restringir a criação e o gerenciamento de credenciais específicas do serviço com base nas configurações de expiração e nos serviços da AWS permitidos. • `iam:ServiceSpecificCredentialAgeDays`	4 de setembro de 2025

	• iam:ServiceSpecificCredentialServiceName	
Chaves de condição de endpoint da VPC para controles de perímetro de rede	O IAM agora oferece suporte a três novas chaves de contexto de condição globais da AWS para implementar controles de perímetro de rede escaláveis: aws:VpceAccount , aws:VpceOrgID , e aws:VpceOrgPaths . Essas chaves ajudam a garantir que as solicitações cheguem por meio de endpoints da VPC pertencentes a contas, organizações ou unidades organizacionais específicas, escalando automaticamente com o uso do seu endpoint da VPC sem exigir atualizações de políticas ao criar novos endpoints. Para obter mais informações, consulte Estabelecer barreiras de proteção para permissões usando perímetros de dados.	28 de agosto de 2025

Tutoriais da federação IAM do SAML do CloudFormation	O IAM adicionou novos tutoriais para criar provedores de identidade SAML (IDPs) e perfis federados usando o CloudFormation. • Criar o IdP do SAML com o CloudFormation • Criar o perfil federado do SAML com o CloudFormation • Criar o IdP do SAML e o perfil federado com o CloudFormation	25 de agosto de 2025
Controles adicionais de provedor OIDC compartilhado para provedores OIDC compartilhados	O IAM agora inclui o Amazon Cognito, o Azure Sentinel, o Pulumi Cloud e o endpoint global do Vercel na lista de provedores de identidades OIDC compartilhados que exigem uma avaliação explícita de declarações específicas em JSON Web Tokens (JWTs).	1.º de agosto de 2025

O IAM Access Analyzer adicionou analisadores de acesso interno	O IAM Access Analyzer ajuda a identificar quais entidades principais da sua organização ou conta têm acesso aos recursos críticos para os negócios selecionados. Os analisadores de acesso interno são compatíveis com a implementação do princípio de privilégio mínimo garantindo que os recursos especificados possam ser acessados somente pelas entidades principais pretendidas da sua organização ou conta da AWS.	16 de junho de 2025
Controles de provedor de identidade para provedores OIDC compartilhados	O IAM agora exige uma avaliação explícita de declarações específicas em JSON Web Tokens (JWTs) para provedores de identidade OIDC compartilhados reconhecidos. Esse controle de segurança garante que somente identidades autorizadas da organização pretendida possam assumir perfis e acessar recursos da AWS.	6 de junho de 2025
`IAMUserChangePassword` – adicionou permissões	O IAM adicionou permissões à política IAMUserChangePassword para permitir que usuários especificados em um determinado caminho possam alterar a senha.	28 de maio de 2025

AccessAnalyzerServiceRolePolicy – adicionou permissões	O IAM Access Analyzer adicionou `iam:GetAccountAuthorizationDetails` às permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	12 de maio de 2025
AccessAnalyzerServiceRolePolicy – adicionou permissões	O Analisador de Acesso do IAM adicionou pontos de acesso de buckets de diretório do Amazon S3 para as permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	31 de março de 2025
Suporte para endpoint com o modo dual stack no IAM	O IAM passou a disponibilizar um suporte aprimorado para endpoint com o modo dual stack, possibilitando que os clientes efetuem comunicações com o IAM por meio tanto de endereços IPv4 quanto de endereços IPv6.	20 de março de 2025
Atualizações na lógica de avaliação de políticas	A documentação da lógica de avaliação de políticas foi aprimorada, incluindo fluxogramas mais detalhados e explicações mais claras sobre o processo da AWS para avaliar se deve permitir ou negar solicitações com base nas políticas.	30 de janeiro de 2025

`IAMDeleteRootUserCredentials` – permissões removidas	O IAM removeu a permissão `iam:DeleteVirtualMFADevice` da política gerenciada.	7 de janeiro de 2025
`AccessAnalyzerServiceRolePolicy` – adicionou permissões	O IAM Access Analyzer adicionou suporte à permissão para recuperar informações sobre políticas de registro e configurações de conta do Amazon ECR para permissões no nível de serviço de AccessAnalyzerServiceRolePolicy.	10 de dezembro de 2024
O IAM Access Analyzer adicionou configuração de acesso	O IAM Access Analyzer adicionou suporte para configurar analisadores para alterar o escopo de quais Contas da AWS, perfis e usuários do IAM geram descobertas.	14 de novembro de 2024

Gerencie centralmente o acesso raiz para contas-membro	Agora é possível gerenciar credenciais de usuário-raiz privilegiado em todas as contas-membro do AWS Organizations com acesso raiz centralizado. Proteja centralmente as credenciais do usuário-raiz das suas Contas da AWS gerenciadas usando o AWS Organizations para remover e impedir a recuperação e o acesso às credenciais de usuário-raiz em grande escala. • Práticas recomendadas do usuário-raiz	14 de novembro de 2024
AWS Atualizações de políticas gerenciadas pela : novas políticas	O IAM adicionou duas novas políticas para definir permissões de escopo para sessões de usuário-raiz privilegiado que podem ser iniciadas depois de centralizar o acesso do usuário-raiz às contas-membro em sua organização. • IAMAuditRootUserCredentials • IAMCreateRootUserPassword • IAMDeleteRootUserCredentials • S3UnlockBucketPolicy • SQSUnlockQueuePolicy	14 de novembro de 2024

Suporte a políticas de controle de recursos (RCPs) do AWS Organizations	Use uma política de controle de recursos (RCP) do AWS Organizations para definir o número máximo de permissões para recursos nas contas da sua organização ou unidade organizacional (UO). As RCPs limitam as permissões que as políticas baseadas em identidade e recursos podem conceder aos recursos nas contas da sua organização.	13 de novembro de 2024
`AccessAnalyzerServiceRolePolicy` – adicionou permissões	O IAM Access Analyzer adicionou suporte à permissão para recuperar informações sobre tags de perfil e usuário do IAM para as permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	29 de outubro de 2024
Aprimoramentos no suporte à criptografia do SAML	A documentação do suporte à criptografia do SAML nos provedores de SAML do IAM foi aprimorada, incluindo diretrizes aprimoradas para solução de problemas e esclarecimentos acerca da compatibilidade com serviços.	5 de junho de 2024

AccessAnalyzerServiceRolePolicy – adicionou permissões	O IAM Access Analyzer adicionou suporte à permissão para recuperar informações sobre políticas de usuário e perfil do IAM para as permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	30 de maio de 2024
Suporte de criptografia a provedores de identidades SAML	Os provedores SAML do IAM agora oferecem suporte a asserções criptografadas na resposta SAML do seu IdP externo. Para entender como a criptografia funciona com a federação SAML do IAM, consulte Usar a federação baseada em SAML para acesso à API.	4 de fevereiro de 2024
AccessAnalyzerServiceRolePolicy – adicionou permissões	O IAM Access Analyzer adicionou suporte à permissão para recuperar o estado atual do bloco de acesso público para instantâneos do Amazon EC2 para as permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	23 de janeiro de 2024
AccessAnalyzerServiceRolePolicy – adicionou permissões	O IAM Access Analyzer adicionou streams e tabelas do DynamoDB às permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	11 de janeiro de 2024

AccessAnalyzerServiceRolePolicy – adicionou permissões	O IAM Access Analyzer adicionou buckets de diretório do Amazon S3 para as permissões de nível de serviço de AccessAnalyzerServiceRolePolicy.	1.º de dezembro de 2023
IAMAccessAnalyzerReadOnlyAccess – adicionou permissões	O IAM Access Analyzer adicionou permissões ao IAMAccessAnalyzerReadOnlyAccess para permitir que você verifique se as atualizações em suas políticas concedem acesso adicional. Essa permissão é exigida pelo IAM Access Analyzer para executar verificações de política em suas políticas.	26 de novembro de 2023
O IAM Access Analyzer adicionou analisadores de acessos não utilizados	O IAM Access Analyzer simplifica a inspeção das acessos não utilizados para guiar você até o privilégio mínimo. O IAM Access Analyzer analisa continuamente suas contas para identificar acessos não utilizados e cria um painel centralizado com as descobertas.	26 de novembro de 2023

O IAM Access Analyzer adicionou verificações de política personalizadas	O IAM Access Analyzer agora fornece verificações de políticas personalizadas para validar se as políticas do IAM estão de acordo com seus padrões de segurança antes das implantações.	26 de novembro de 2023
`AccessAnalyzerServiceRolePolicy` – adicionou permissões	O IAM Access Analyzer adicionou ações do IAM para as permissões no nível do serviço da AccessAnalyzerServiceRolePolicy para dar suporte para as seguintes ações: • Listando entidades para uma política • Gerando detalhes do serviço acessados pela última vez • Listar informações de chave de acesso	26 de novembro de 2023
Compatibilidade com informações sobre a última ação acessada e com geração de políticas em mais de 60 serviços e ações adicionais	O IAM agora é compatível com informações sobre a última ação acessada e gera políticas com informações no nível da ação para mais de 60 serviços adicionais, juntamente com uma lista das ações para as quais as informações da última ação acessada estão disponíveis.	1º de novembro de 2023

Compatibilidade com informações sobre a última ação acessada em mais 140 serviços	O IAM agora fornece informações sobre a última ação acessada em mais de 140 serviços, juntamente com uma lista das ações para as quais as informações da última ação acessada estão disponíveis.	14 de setembro de 2023
Suporte para dispositivos com autenticação multifator (MFA) para usuários raiz e usuários do IAM	Agora você pode adicionar até oito dispositivos de MFA por usuário, inclusive chaves de segurança FIDO, senha de uso único com marcação temporal (TOTP) com aplicações de autentica dores virtuais ou tokens TOTP de hardware.	16 de novembro de 2022
Suporte do IAM Access Analyzer para novos tipos de recursos	O IAM Access Analyzer adicionou suporte para os tipos de recursos a seguir: • Snapshots de volume do Amazon EBS • Repositórios do Amazon ECR • Sistemas de arquivos do Amazon EFS • Snapshots de banco de dados do Amazon RDS • Snapshots de cluster de banco de dados do Amazon RDS • Tópicos do Amazon SNS	25 de outubro de 2022

Descontinuação de U2F e atualização da WebAuthn/FIDO	Removidas menções à U2F como uma opção de MFA e adicionadas informações sobre as chaves de segurança WebAuthn, FIDO2 e FIDO.	31 de maio de 2022
Atualizações da resiliência no IAM	Foram adicionadas informaçõ es sobre como manter o acesso às credenciais do IAM quando um evento interrompe a comunicação entre Regiões da AWS.	16 de maio de 2022
Novas chaves de condições globais para recursos	Agora você pode controlar o acesso a recursos baseados na conta, Unidade Organizac ional (UO) ou organizaç ão no AWS Organizations que contém os recursos. É possível usar as chaves de condições globais `aws:Resou rceAccount` , `aws:Resou rceOrgID` e `aws:Resou rceOrgPaths` em uma política do IAM.	27 de abril de 2022

Exemplos de código do IAM que usam AWS SDKs	Os exemplos de código a seguir mostram como usar o IAM com um Kit de Desenvolvimento de Software (SDK) da AWS. Os exemplos são divididos em trechos de código que mostram como chamar funções de serviço individuais e exemplos que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.	7 de abril de 2022
Atualizações do fluxograma de lógica de avaliação de política	Atualizações no fluxograma de lógica de avaliação de política e do texto relacionado na seção Determinar se uma solicitação é permitida ou negada em uma conta.	17 de novembro de 2021
Atualizações das práticas recomendadas de segurança	Adicionadas informações sobre criar usuários administrativos em vez usar credenciais de usuário-raiz, removida a prática recomendada de usar grupos do IAM para atribuir permissões a usuários do IAM e esclarecido quando usar políticas gerenciadas em vez de políticas em linha.	5 de outubro de 2021
Atualizações do tópico de lógica de avaliação de política para políticas baseadas em recursos	Adição de informações sobre o impacto de políticas baseadas em recursos e diferentes tipos de entidade principal na mesma conta.	5 de outubro de 2021

Atualizações das chaves de condição com valor único e com valores múltiplos	Agora as diferenças entre chaves de condição de valor único e de valores múltiplos estão explicadas com mais detalhes. O tipo de valor foi adicionado a cada chave de contexto de condição global da AWS.	30 de setembro de 2021
O IAM Access Analyzer é compatível com pontos de acesso multirregionais do Amazon S3	O IAM Access Analyzer identifica buckets do Amazon S3 que permitem acesso público e entre contas, incluindo aqueles que usam pontos de acesso multirregiões do Amazon S3.	2 de setembro de 2021
AWS Atualizações de política gerenciada pela : atualização para uma política existente	O IAM Access Analyzer atualizou uma política existente gerenciada pela AWS.	2 de setembro de 2021
Mais serviços com suporte para a geração de políticas no nível da ação	O IAM Access Analyzer pode gerar políticas do IAM com informações de atividade de acesso no nível da ação para produtos da AWS adicionais.	24 de agosto de 2021
Gerar políticas do IAM para trilhas entre contas	Agora você pode usar o IAM Access Analyzer para gerar políticas refinadas com base em sua atividade de acesso usando uma trilha do AWS CloudTrail em uma conta diferente, por exemplo, uma trilha do AWS Organizations centralizada.	18 de agosto de 2021

Verificações de política adicionais do IAM Access Analyzer	O IAM Access Analyzer estendeu a validação de políticas adicionando novas verificações de política que validam as condições incluídas nas políticas do IAM. Essas verificações analisam o bloco de condições em sua instrução de política e relatam avisos, erros e sugestões de segurança, juntamente com recomendações práticas. O IAM Access Analyzer adicionou as seguintes verificações de política: • Erro: Invalid service principal format (Formato de entidade de serviço inválido • Erro: Missing tag key in condition (Chave de etiqueta ausente na condição • Aviso de segurança: Deny NotAction with unsupported tag condition key for service (Negar NotAction com chave de condição de etiqueta não suportada para o serviço • Aviso de segurança: Deny with unsupported tag condition key for service (Negar com chave de condição de etiqueta não suportada para o serviço	29 de junho de 2021

- Aviso de segurança: Missing paired condition keys (Chaves de condição emparelhadas ausentes
- Sugestão: Allow NotAction with unsupported tag condition key for service (Permitir NotAction com chave de condição de etiqueta não suportada para o serviço
- Sugestão: Allow with unsupported tag condition key for service (Permitir com chave de condição de etiqueta sem suporte para o serviço

Suporte de último acesso da ação para mais serviços	Agora você pode ver as informações do último acesso da ação no console do IAM sobre a última vez que uma entidade do IAM usou uma ação para os seguintes serviços: ações de gerenciamento do Amazon EC2, IAM, Lambda e Amazon S3. Você também pode usar a AWS CLI ou a API da AWS para recuperar um relatório de dados. É possível usar essas informações para identificar permissões desnecessárias, de forma que você possa refinar suas políticas do IAM para melhor aderir ao princípio de privilégio mínimo.	19 de abril de 2021
Monitorar e controlar ações realizadas com funções assumidas	Os administradores podem configurar funções do IAM para exigir que as identidades passem uma identidade-fonte, que está conectada ao AWS CloudTrail. A revisão das informações de identidade-fonte ajuda os administradores a determinar quem ou o que executou ações com sessões de função assumida.	13 de abril de 2021

Gerar políticas do IAM com base na atividade de acesso	Agora você pode usar o IAM Access Analyzer para gerar políticas refinadas com base em sua atividade de acesso encontrada no seu AWS CloudTrail.	7 de abril de 2021
Verificações de política do IAM Access Analyzer	O IAM Access Analyzer agora fornece mais de 100 verificações de políticas com recomendações práticas durante a criação de políticas.	16 de março de 2021
Opções de validação de políticas expandidas	Validação de políticas expandidas disponível no console do IAM, na API da AWS e na AWS CLI usando verificações de política no IAM Access Analyzer para ajudar você a criar políticas de JSON seguras e funcionais.	15 de março de 2021
Recursos de etiquetas do IAM	Agora você pode etiquetar recursos adicionais do IAM usando um par de chave-valor de etiqueta.	11 de fevereiro de 2021
Política de senha padrão para usuários do IAM	Se você não definir uma política de senha personali zada para sua Conta da AWS, as senhas do usuário do IAM agora deverão atender à política de senha padrão da AWS.	18 de novembro de 2020

As páginas de ações, recursos e chaves de condições dos serviços da AWS foram movidas	Cada serviço da AWS pode definir ações, recursos e chaves de contexto de condição para uso em políticas do IAM. Agora você pode encontrar a lista de serviços AWS e suas ações, recursos e chaves de contexto de condição na Referência de autorização de serviço.	16 de novembro de 2020
Maior duração da sessão da função de usuários do IAM	Os usuários do IAM agora podem ter uma duração da sessão da função mais longa ao mudar de funções no Console de gerenciamento da AWS, reduzindo interrupç ões devido à expiração da sessão. Os usuários recebem a duração máxima da sessão definida para a função ou o tempo restante na sessão do usuário do IAM, o que for menor.	24 de julho de 2020
Usar o Service Quotas para solicitar aumentos rápidos para entidades do IAM	É possível solicitar aumentos de cota para cotas ajustáveis do IAM usando o console do Service Quotas. Agora, alguns aumentos são automatic amente aprovados no Service Quotas e disponibilizados em sua conta em poucos minutos. Solicitações maiores são enviadas ao AWS Support.	25 de junho de 2020

As informações de último acesso no IAM agora incluem ações de gerenciamento do Amazon S3	Além das informações do último acesso do serviço, agora você pode visualizar informações no console do IAM sobre a última vez que uma entidade do IAM usou uma ação do Amazon S3. Você também pode usar a AWS CLI ou a API da AWS para recuperar o relatório de dados. O relatório inclui informações sobre os serviços permitidos e ações que os principais tentaram acessar pela última vez e quando. É possível usar essas informaçõ es para identificar permissõe s desnecessárias, de forma que você possa refinar suas políticas do IAM para melhor aderir ao princípio de privilégio mínimo.	3 de junho de 2020
Adição de capítulo sobre segurança	O capítulo de segurança ajuda a entender como configura r o IAM e o AWS STS para atender aos seus objetivos de segurança e conformid ade. Saiba também como usar outros produtos da AWS que ajudam a monitorar e proteger os recursos do IAM.	29 de abril de 2020

sts:RoleSessionName	Agora você pode escrever uma política que concede permissões com base no nome da sessão que um principal especifica ao assumir uma função.	21 de abril de 2020
AWS Atualização da página de login da	Ao fazer login na página principal da AWS, agora não será possível optar por fazer login como o Usuário raiz da conta da AWS ou como um usuário do IAM. Ao fazer isso, o rótulo na página indica se você deve fornecer o endereço do e-mail do usuário-raiz ou as informaçõ es do usuário do IAM. Esta documentação inclui capturas de telas atualizadas para ajudar você a compreender as páginas de login da AWS.	4 de março de 2020

Chaves de condições aws:ViaAWSService e aws:CalledVia	Agora você pode criar uma política para limitar se os serviços podem fazer solicitações em nome de uma entidade do IAM (usuário ou função). Quando um principal faz uma solicitação a um serviço da AWS, esse serviço pode usar as credenciais do principal para fazer solicitações subsequentes a outros serviços. Use a chave de condição `aws:ViaAWSService` para corresponder se algum serviço fizer uma solicitação usando as credenciais de um principal. Use as chaves de condição `aws:CalledVia` para corresponder se serviços específicos fizerem uma solicitação usando as credenciais de um principal.	20 de fevereiro de 2020
O simulador de políticas adiciona compatibilidade com limites de permissões	Agora você pode testar o efeito dos limites de permissões em entidades do IAM com o simulador de políticas do IAM.	23 de janeiro de 2020

Avaliação de políticas entre contas	Agora é possível saber como a AWS avalia as políticas para acesso entre contas. Isso ocorre quando um recurso em uma conta confiável inclui uma política baseada em recursos que permite que um principal em outra conta acesse o recurso. A solicitação deve ser permitida em ambas as contas.	2 de janeiro de 2020
Tags de sessão	Agora você pode incluir tags ao assumir uma função ou agrupar um usuário no AWS STS. Ao executar a operação `AssumeRole` ou `GetFederationToken`, você pode passar as tags de sessão como atributos. Ao executar as operações `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity`, você pode passar atributos de suas identidades corporativas para a AWS.	22 de novembro de 2019

Controle de acesso para grupos de Contas da AWS no AWS Organizations	Agora você pode referenciar unidades organizacionais (UOs) do AWS Organizations nas políticas do IAM. Se usar AWS Organizations para organizar suas contas em UOs, você poderá exigir que os principais pertençam a uma UO específica antes de conceder acesso aos seus recursos. As entidades principais incluem Usuário raiz da conta da AWS, usuários do IAM e perfis do IAM. Para isso, especifique o caminho da UO na chave de condição `aws:PrincipalOrgPaths` nas suas políticas.	20 de novembro de 2019
Última função usada	Agora você pode exibir a data, hora e região onde uma função foi usada pela última vez. Essas informações também ajudam a identificar as funções não utilizadas na sua conta. É possível usar o Console de gerenciamento da AWS, a AWS CLI e a API da AWS para exibir informações sobre quando um perfil foi usado pela última vez.	19 de novembro de 2019

Atualização da página de chaves contextuais de condições globais	Agora é possível saber quando cada uma das chaves de condição globais está incluída no contexto de uma solicitação. Também é possível navegar até cada chave com mais facilidad e usando o índice (TOC) da página. As informaçõ es na página ajudam você a escrever políticas mais precisas. Por exemplo, se os seus funcionários usarem federação com funções do IAM, você deverá usar a chave `aws:userId` e não a chave `aws:userName` . A chave `aws:userName` se aplica apenas a usuários do IAM, e não a funções.	6 de outubro de 2019

ABAC na AWS	Saiba como o controle de acesso baseado em atributo (ABAC) funciona na AWS usando tags e como ele se compara ao modelo de autorização tradicional da AWS. Use o tutorial do ABAC para saber como criar e testar uma política que permite que funções do IAM com etiquetas de entidade acessem recursos com etiquetas correspon dentes. Essa estratégia permite que os indivíduos visualizem ou editem apenas os recursos da AWS necessári os para seus trabalhos.	3 de outubro de 2019
AWS STS Operação GetAccessKeyInfo do	É possível revisar as chaves de acesso da AWS em seu código para determinar se as chaves são de uma conta que você possui. É possível transmitir um ID de chave de acesso usando o comando `aws sts get-access-key-info` da AWS CLI ou a operação da API `GetAccessKeyInfo` da AWS.	24 de julho de 2019

Visualizar informações do serviço do AWS Organizations acessadas pela última vez no IAM	Agora você pode visualizar informações do último acesso do serviço de uma entidade ou política do AWS Organizations na seção AWS Organizations do console do IAM. Você também pode usar a AWS CLI ou a API da AWS para recuperar o relatório de dados. Esses dados incluem informações sobre quais serviços permitidos as entidades principais em uma conta do AWS Organizations tentaram acessar mais recentemente e quando. Você pode usar esta informação para identificar permissões desnecessárias, de forma que você possa refinar suas políticas do AWS Organizations para melhor aderir ao princípio de privilégio mínimo.	20 de junho de 2019
Usar uma política gerenciada como uma política de sessão	Agora, você pode transmitir até 10 ARNs de políticas gerenciadas ao assumir uma função. Isso permite que você limite as permissões das credenciais temporárias da função.	7 de maio de 2019

AWS STS Compatibilidade de regiões do de tokens de sessão para o endpoint global	Agora, você pode escolher se deseja usar tokens do endpoint global de versão 1 ou versão 2. Tokens de versão 1 são válidos somente em regiões da AWS que estão disponíveis por padrão. Esses tokens não funcionar ão em regiões habilitadas manualmente, como Ásia-Pací fico (Hong Kong). Tokens de versão 2 são válidos em todas as regiões. No entanto, tokens de versão 2 são maiores e podem afetar sistemas em que você armazena tokens temporariamente.	26 de abril de 2019
Permitir habilitar e a desabilitar regiões da AWS	Agora, você pode criar uma política que permite que um administrador habilite e desabilite a região Ásia-Pací fico (Hong Kong) (ap-east-1).	24 de abril de 2019

Página My security credentials (Minhas credenciais de segurança) do usuário do IAM	Os usuários do IAM agora podem gerenciar as próprias credenciais na página My Security Credentials (Minhas credenciais de segurança). Essa página do Console de gerenciamento da AWS exibe as informações de conta, como ID de conta e ID de usuário canônico. Os usuários também podem visualizar e editar as próprias senhas, chaves de acesso, certificados X.509, chaves SSH e credenciais do Git.	24 de janeiro de 2019
API do Access Analyzer	Você já pode usar a AWS CLI e a API da AWS para visualizar informações de serviço acessadas por último.	7 de dezembro de 2018
Etiquetar usuários e funções do IAM	Agora é possível usar tags do IAM para adicionar atributos personalizados a uma identidade (usuário ou função do IAM) usando um par de chave-valor de tag. Você também pode usar tags para controlar o acesso de uma identidade a recursos ou para controlar quais tags podem ser anexadas a uma identidade.	14 de novembro de 2018

Chaves de segurança U2F	Agora, você pode usar chaves de segurança U2F como uma opção de autenticação multifator (MFA) quando fizer login no Console de gerenciamento da AWS.	25 de setembro de 2018
Suporte para endpoints da Amazon VPC	Agora, você pode estabelecer uma conexão privada entre a VPC e o AWS STS na região Oeste dos EUA (Oregon).	31 de julho de 2018
Limites de permissões	O novo recurso facilita conceder aos funcionários confiáveis a capacidade de gerenciar permissões do IAM sem também conceder acesso administrativo completo do IAM.	12 de julho de 2018
aws:PrincipalOrgID	A nova chave de condição fornece uma maneira mais fácil de controlar o acesso a recursos da AWS especific ando a organização da AWS das entidades do IAM.	17 de maio de 2018
aws:RequestedRegion	A nova chave de condição fornece uma forma mais fácil de usar as políticas do IAM para controlar o acesso às regiões da AWS.	25 de abril de 2018
Maior duração da sessão para funções do IAM	Uma função do IAM agora pode ter uma sessão com duração de 12 horas.	28 de março de 2018

Fluxo de trabalho de criação de funções atualizado	O novo fluxo de trabalho melhora o processo de criação de relações de confiança e anexação de permissões a funções.	8 de setembro de 2017
Conta da AWS Processo de login na	A experiência de login da AWS atualizada permite que o usuário-raiz e os usuários do IAM usem o link Sign In to the Console na página inicial do Console de gerenciamento da AWS.	25 de agosto de 2017
Políticas de exemplo do IAM	A atualização da documenta ção apresenta mais de 30 políticas de exemplo.	2 de agosto de 2017
Práticas recomendadas do IAM	As informações adicionad as à seção Users (Usuários) do console do IAM facilitam a adoção das práticas recomendadas do IAM.	5 de julho de 2017
Recursos de dimension amento automático	As permissões em nível de recurso podem controlar o acesso aos recursos do Auto Scaling e suas permissões.	16 de maio de 2017
Bancos de dados do Amazon RDS for MySQL e Amazon Aurora	Os administradores de banco de dados podem associar usuários de banco de dados com usuários e funções do IAM e, portanto, gerenciar o acesso dos usuários a todos os recursos da AWS de um único local.	24 de abril de 2017

Perfis vinculados ao serviço	As funções vinculadas ao serviço fornecem uma maneira mais fácil e mais segura para delegar permissões para serviços da AWS.	19 de abril de 2017
Resumos de políticas	Novos resumos de políticas facilitam a compreensão das permissões em políticas do IAM.	23 de março de 2017